数据接入服务

API 参考

文档版本 02

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

欢迎使用数据接入服务（Data Ingestion Service, DIS）。数据接入服务面向IoT、互联网等实时数据，提供高效采集、传输、分发能力，支持多种IoT协议，提供丰富的接口，帮助您快速构建实时数据应用。

您可以使用本文档提供API对实时数据进行相关操作，如上传、下载等。支持的全部操作请参见**API概览**。

在调用数据接入服务API之前，请确保已经充分了解数据接入服务相关概念，详细信息请参见“**产品介绍**”。

1.2 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，而是创建用户并使用他们进行日常管理工作。

- 用户

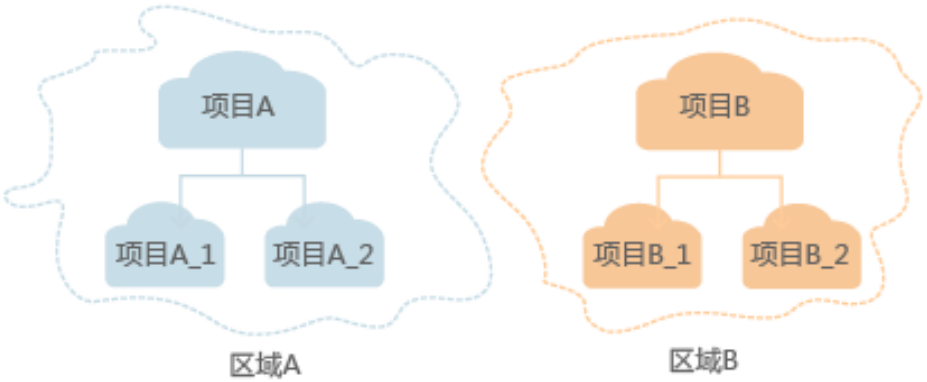
  由账号在IAM中创建的用户，是云服务的使用人员，具有身份凭证（密码和访问密钥）。

  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，并在子项目中创建资源，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



- Checkpoint

  消费检查点。应用程序消费数据时，记录已消费数据的最新序列号作为检查点。当重新消费数据时，可根据此检查点继续消费。

- APP

  应用程序标识符。当多个应用程序分别消费同一通道的数据时，为区分不同应用程序的消费检查点，使用APP作为标识。

2 API 概览

DIS提供的接口为符合RESTful API设计规范的自研接口。

通过DIS的自研接口，您可以使用DIS的如**API说明**所示的功能。

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**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。

请求 URI

请求URI由如下部分组成：

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

尽管请求URI包含在请求消息头中，但大多数语言或框架都要求您从请求消息中单独传递它，所以在此单独强调。

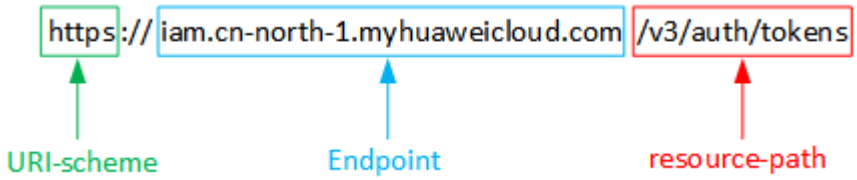
表 3-1 URI 中的参数说明

参数	描述
URI-scheme	表示用于传输请求的协议，当前所有API均采用**HTTPS**协议。
Endpoint	指定承载REST服务端点的服务器域名或IP，不同服务不同区域的Endpoint不同，您可以从**地区和终端节点**中获取。 例如IAM服务在“华北-北京四”区域的Endpoint为“iam.cn-north-4.myhuaweicloud.com”。
resource-path	资源路径，也即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，是可选部分，并不是每个API都有查询参数。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IAM在“华北-北京四”区域的Token，则需使用“华北-北京四”区域的Endpoint（iam.cn-north-4.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-4.myhuaweicloud.com/v3/auth/tokens
```

图 3-1 URI 示意图



说明

为查看方便，在每个具体API的URI部分，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），它告诉服务你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-4.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。

详细的公共请求消息头字段请参见**表3-3**。

表 3-3 公共请求消息头

名称	描述	是否必选	示例
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 使用AK/SK认证时该字段必选。	code.test.com or code.test.com:443
Content-Type	发送的实体的MIME类型。推荐用户默认使用application/json，有其他取值时会在具体接口中专门说明。	是	application/json
Content-Length	请求body长度，单位为Byte。	否	3495
X-Project-Id	project id，项目编号。请参考**获取项目ID**章节获取项目编号。	否	e9993fc787d94b6c886cbaa340f9c0f4
X-Auth-Token	用户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。 请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。	否 使用Token认证时该字段必选。	注：以下仅为Token示例片段 MIIPAgYJKoZIhvcNAQcCo...ggg1BBIINPXsidG9rZ

说明

API同时支持使用AK/SK认证，AK/SK认证是使用SDK对请求进行签名，签名过程会自动往请求中添加Authorization（签名认证信息）和X-Sdk-Date（请求发送的时间）请求头。

AK/SK认证的详细说明请参见**认证鉴权**的“AK/SK认证”。

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-4.myhuaweicloud.com/v3/auth/tokens
Content-Type: application/json
```

请求消息体（可选）

该部分可选。请求消息体通常以结构化格式（如JSON或XML）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。若请求消息体中的参数支持中文，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，加粗的斜体字段需要根据实际值填写，其中***username***为用户名，***domainname***为用户所属的账号名称，*********为用户登录密码，***xxxxxxxxxxxxxxxxx***为project的ID，获取方法请参见**获取项目ID**。

说明

scope参数定义了Token的作用域，下面示例中获取的Token仅能访问project下的资源。您还可以设置Token的作用域为某个账号下所有资源或账号的某个project下的资源，详细定义请参见**获取用户Token**。

```
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": {
                "name": "xxxxxxxxxxxxxxxxx"
            }
        }
    }
}
```

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”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.ap-southeast-1.myhuaweicloud.com/v1/{project_id}/connections
Content-Type: application/json
X-Auth-Token: ABCDEFG....
```

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。

3.3 返回结果

状态码

请求发送以后，您会收到响应，包含状态码、响应消息头和消息体。

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”。

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

响应消息体（可选）

该部分可选。响应消息体通常以结构化格式（如JSON或XML）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": "******",
......
```

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。

4 应用示例

操作场景

数据接入服务面向实时数据，提供高效采集、传输、分发能力，提供丰富的接口，帮助您快速构建实时数据应用。

下面介绍如何调用**创建通道**API创建数据接入通道，API的调用方法请参见**如何调用API**。

说明

通过IAM服务获取到的Token有效期为24小时，需要使用同一个Token鉴权时，可以先将Token缓存，避免频繁调用。

涉及 API

当您使用Token认证方式完成认证鉴权时，需要获取用户Token并在调用接口时增加“X-Auth-Token”到业务接口请求消息头中。

- IAM获取token的API
- DIS创建通道的API

前提条件

您需要规划数据接入服务所在的区域信息，并根据区域确定调用API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。

创建通道

如下示例是创建通道最简单的配置。

1. Token认证，具体操作请参考**Token认证**。
2. 发送“POST https://dis的Endpoint/v2/{project_id}/streams”。
3. 在Request Header中增加“X-Auth-Token”。
4. 在Request Body中传入参数如下：

```
{
"stream_name": "dis-DLpR",
"partition_count": 1,
```

```
"stream_type": "COMMON",
"data_duration": 24
}
```

- stream_name：通道的名称，由您自行定义，例如取名为newstream。
- partition_count：分区是DIS数据通道的基本吞吐量单位，您可以根据业务吞吐的需求选择通道的分区数。
- stream_type：通道类型，“COMMON”表示普通分区，单分区支持最大1MB/s的写入速度和2MB/s的读取速度。
- data_duration：通道生命周期，即通道分区中数据的保留时长。

请求响应成功后，返回201 Created，表示通道创建成功。

若请求失败，则会返回错误码及对应的错误信息说明，详细错误码信息请参考**错误码**。

创建支持自动扩缩容的通道

您还可以创建支持自动扩缩容的通道，可以根据通道的流量情况自动为您扩充或缩减分片数量。示例如下。

1. Token认证，具体操作请参考**Token认证**。
2. 发送“POST https://dis的Endpoint/v2/{project_id}/streams”。
3. 在Request Header中增加“X-Auth-Token”。
4. 在Request Body中传入参数如下：

```
{
"stream_name": "dis-DLpR",
"partition_count": 1,
"stream_type": "COMMON",
"data_duration": 24
"auto_scale_enabled": true,
"auto_scale_min_partition_count": 2,
"auto_scale_max_partition_count": 10
}
```

这个示例中创建了支持自动扩缩容的通道，通道缩容的最小分区数量是2，扩容的最大分区数量是10，也就是说如果通道10个分区后不会再触发自动扩容。

- auto_scale_enabled：是否开启自动扩缩容，true表示开启。
- auto_scale_min_partition_count：当自动扩缩容启用时，自动缩容的最小分片数，比如这个示例中当分区数量为2时，不会再触发自动缩容。
- auto_scale_max_partition_count：当自动扩缩容启用时，自动扩容的最大分片数，比如这个示例中当分区数量为10时，不会再触发自动扩容。

请求响应成功后，返回201 Created，表示通道创建成功。

若请求失败，则会返回错误码及对应的错误信息说明，详细错误码信息请参考**错误码**。

创建带数据 Schema 的通道

您还可以为通道配置Schema，在使用DIS转储到其它服务时，可以根据通道配置的Schema来完成映射，示例如下。

1. Token认证，具体操作请参考**Token认证**。
2. 发送“POST https://dis的Endpoint/v2/{project_id}/streams”。
3. 在Request Header中增加“X-Auth-Token”。
4. 在Request Body中传入参数如下：


```
{
"stream_name": "dis-DLpR",
"partition_count": 1,
"stream_type": "COMMON",
"data_duration": 24
"auto_scale_enabled": true,
"auto_scale_min_partition_count": 1,
"auto_scale_max_partition_count": 10
"data_type": "JSON",
"data_schema":
"{\"type\":\"record\",\"name\":\"RecordName\",\"fields\":[{\"name\":\"key1\",\"type\":\"string
\"},{\"name\":\"key2\",\"type\":\"string\"}]}"
}
```

这个示例中创建了一个源数据类型为JSON，且数据包含“key1”、“key2”这两个属性的通道。

- data_type：指定源数据的类型，“JSON”表示分区中的数据格式为JSON格式。
- data_schema：源数据Schema，用于描述JSON、CSV格式的源数据结构，采用Avro Schema的语法描述。

请求响应成功后，返回201 Created，表示通道创建成功。

若请求失败，则会返回错误码及对应的错误信息说明，详细错误码信息请参考**错误码**。

5 API 说明

5.1 通道管理

5.1.1 创建通道

功能介绍

本接口用于创建通道。

- 创建通道时，需指定通道类型（普通、高级）、分区数量。
- 一个账号默认最多可以创建10个高级通道分区和50个普通通道分区，可提交工单增加配额。

调用方法

请参见**如何调用API**。

URI

POST /v2/{project_id}/streams

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。

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）。

表 5-3 请求 Body 参数

参数	是否必选	参数类型	描述
stream_name	是	String	通道名称。 通道名称由字母、数字、下划线和中划线组成，长度为1～64字符。
partition_count	是	Integer	分区数量。 分区是DIS数据通道的基本吞吐量单位。
stream_type	否	String	通道类型。 ● COMMON：普通通道，表示1MB带宽。 ● ADVANCED：高级通道，表示5MB带宽。
data_type	否	String	源数据类型。 ● BLOB：存储在数据库管理系统中的一组二进制数据。 ● JSON：一种开放的文件格式，以易读的文字为基础，用来传输由属性值或者序列性的值组成的数据对象。 ● CSV：纯文本形式存储的表格数据，分隔符默认采用逗号。 缺省值：BLOB。
data_duration	否	Integer	数据保留时长。 取值范围：24~72。 单位：小时。 空表示使用缺省值。

参数	是否必选	参数类型	描述
auto_scale_enabled	否	Boolean	是否开启自动扩缩容。 ● true：开启自动扩缩容。 ● false：关闭自动扩缩容。 默认不开启。
auto_scale_min_partition_count	否	Long	当自动扩缩容启用时，自动缩容的最小分片数。
auto_scale_max_partition_count	否	Integer	当自动扩缩容启用时，自动扩容的最大分片数。
data_schema	否	String	用于描述用户JSON、CSV格式的源数据结构，采用Avro Schema的语法描述。
csv_properties	否	**CSVProperties** object	CSV 格式数据的相关属性，比如分隔符 delimiter
compression_format	否	String	数据的压缩类型，目前支持： ● snappy ● gzip ● zip 默认不压缩。
tags	否	Array of **Tag** objects	通道标签列表。
sys_tags	否	Array of **SysTag** objects	通道企业项目列表。

表 5-4 CSVProperties

参数	是否必选	参数类型	描述
delimiter	否	String	数据分隔符。

表 5-5 Tag

参数	是否必选	参数类型	描述
key	否	String	键。 ● 不能为空。 ● 对于同一资源键值唯一。 ● 字符集：A-Z，a-z ， 0-9，‘-’，‘_’，UNICODE字符（\u4E00-\u9FFF）。
value	否	String	值。 ● 长度不超过43个字符。 ● 字符集：A-Z，a-z ， 0-9，‘.’，‘-’，‘_’，UNICODE字符（\u4E00-\u9FFF）。 ● 只能包含数字、字母、中划线“-”、下划线“_”。

表 5-6 SysTag

参数	是否必选	参数类型	描述
key	否	String	键。 ● 不能为空。 ● 值必须为_sys_enterprise_project_id。
value	否	String	值。 ● 对应的是企业项目ID，需要在企业管理页面获取。 ● 36位UUID。

响应参数

无

请求示例

创建通道

```
POST https://{Endpoint}/v2/{project_id}/streams

{
  "stream_name" : "newstream",
  "partition_count" : 3,
  "data_duration" : 24
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

创建通道

```
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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class CreateStreamSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateStreamRequest request = new CreateStreamRequest();
        CreateStreamReq body = new CreateStreamReq();
        body.withDataDuration(24);
        body.withPartitionCount(3);
        body.withStreamName("newstream");
        request.withBody(body);
        try {
            CreateStreamResponse response = client.createStream(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

创建通道

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateStreamRequest()
        request.body = CreateStreamReq(
            data_duration=24,
            partition_count=3,
            stream_name="newstream"
        )
        response = client.create_stream(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

创建通道

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateStreamRequest{}
    dataDurationCreateStreamReq:= int32(24)
    request.Body = &model.CreateStreamReq{
        DataDuration: &dataDurationCreateStreamReq,
        PartitionCount: int32(3),
        StreamName: "newstream",
    }
    response, err := client.CreateStream(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
201	Created

错误码

请参见**错误码**。

5.1.2 查询通道列表

功能介绍

本接口用户查询当前租户创建的所有通道。

查询时，需要指定从哪个通道开始返回通道列表和单次请求需要返回的最大数量。

调用方法

请参见**如何调用API**。

URI

GET /v2/{project_id}/streams

表 5-7 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

表 5-8 Query 参数

参数	是否必选	参数类型	描述
limit	否	Integer	单次请求返回通道列表的最大数量。
start_stream_name	否	String	从该通道开始返回通道列表，返回的通道列表不包括此通道名称。如果需要分页查询，第一页查询时不传该字段。返回结果has_more_streams为true时，进行下一页查询，start_stream_name传入第一页查询结果的最后一条通道名称。

请求参数

表 5-9 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 5-10 响应 Body 参数

参数	参数类型	描述
total_number	Long	当前租户所有通道数量。
stream_names	Array of strings	满足当前请求条件的通道名称的列表。

参数	参数类型	描述
has_more_streams	Boolean	是否还有更多满足条件的通道。 ● true：是 ● false：否
stream_info_list	Array of **StreamInfo** objects	通道列表详情。

表 5-11 StreamInfo

参数	参数类型	描述
stream_name	String	通道名称。
create_time	Long	通道创建的时间，13位时间戳。
retention_period	Integer	数据保留时长，单位是小时。
status	String	通道的当前状态。 ● CREATING：创建中 ● RUNNING：运行中 ● TERMINATING：删除中 ● TERMINATED：已删除
stream_type	String	通道类型。 ● COMMON：普通通道，表示1MB带宽。 ● ADVANCED：高级通道，表示5MB带宽。
data_type	String	源数据类型。 ● BLOB：存储在数据库管理系统中的一组二进制数据。 ● JSON：一种开放的文件格式，以易读的文字为基础，用来传输由属性值或者序列性的值组成的数据对象。 ● CSV：纯文本形式存储的表格数据，分隔符默认采用逗号。 缺省值：BLOB。
partition_count	Integer	分区数量。 分区是DIS数据通道的基本吞吐量单位。

参数	参数类型	描述
auto_scale_enabled	Boolean	是否开启自动扩缩容。 ● true：开启自动扩缩容。 ● false：关闭自动扩缩容。 默认不开启。
auto_scale_min_partition_count	Integer	当自动扩缩容启用时，自动缩容的最小分片数。
auto_scale_max_partition_count	Integer	当自动扩缩容启用时，自动扩容的最大分片数。
tags	Array of **Tag** objects	通道标签列表。
sys_tags	Array of **SysTag** objects	通道企业项目列表。

表 5-12 Tag

参数	参数类型	描述
key	String	键。 ● 不能为空。 ● 对于同一资源键值唯一。 ● 字符集：A-Z，a-z ， 0-9， ‘-’，‘_’，UNICODE字符（\u4E00-\u9FFF）。
value	String	值。 ● 长度不超过43个字符。 ● 字符集：A-Z，a-z ， 0-9， ‘.’，‘-’ ， ‘_’，UNICODE字符（\u4E00-\u9FFF）。 ● 只能包含数字、字母、中划线“-”、下划线“_”。

表 5-13 SysTag

参数	参数类型	描述
key	String	键。 ● 不能为空。 ● 值必须为_sys_enterprise_project_id。

参数	参数类型	描述
value	String	值。 ● 对应的是企业项目ID，需要在企业管理页面获取。 ● 36位UUID。

请求示例

查询通道列表

```
GET https://{Endpoint}/v2/{project_id}/streams
```

响应示例

状态码：200

正常返回

```
{
  "total_number" : 1,
  "stream_names" : [ "newstream" ],
  "stream_info_list" : [ {
    "stream_id" : "8QM3Nt9YTLOwtUVYJhO",
    "stream_name" : "newstream",
    "create_time" : 1593569685875,
    "retention_period" : 24,
    "status" : "RUNNING",
    "stream_type" : "COMMON",
    "data_type" : "BLOB",
    "partition_count" : 1,
    "tags" : [ ],
    "auto_scale_enabled" : false,
    "auto_scale_min_partition_count" : 0,
    "auto_scale_max_partition_count" : 0
  } ],
  "has_more_streams" : false
}
```

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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class ListStreamsSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        ListStreamsRequest request = new ListStreamsRequest();
        try {
            ListStreamsResponse response = client.listStreams(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 huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListStreamsRequest()
        response = client.list_streams(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"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListStreamsRequest{}
    response, err := client.ListStreams(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	正常返回

错误码

请参见**错误码**。

5.1.3 删除指定通道

功能介绍

本接口用于删除指定通道。

调用方法

请参见**如何调用API**。

URI

DELETE /v2/{project_id}/streams/{stream_name}

表 5-14 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
stream_name	是	String	需要删除的通道名称。

请求参数

表 5-15 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

无

请求示例

删除指定通道

```
DELETE https://{Endpoint}/v2/{project_id}/streams/{stream_name}
```

响应示例

无

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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class DeleteStreamSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        DeleteStreamRequest request = new DeleteStreamRequest();
        request.withStreamName("{stream_name}");
        try {
            DeleteStreamResponse response = client.deleteStream(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 huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteStreamRequest()
        request.stream_name = "{stream_name}"
        response = client.delete_stream(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"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteStreamRequest{}
    request.StreamName = "{stream_name}"
    response, err := client.DeleteStream(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
204	正常返回

错误码

请参见**错误码**。

5.1.4 查看通道详情

功能介绍

本接口用于查询指定通道的详情。

调用方法

请参见**如何调用API**。

URI

GET /v2/{project_id}/streams/{stream_name}

表 5-16 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
stream_name	是	String	需要查询的通道名称。

表 5-17 Query 参数

参数	是否必选	参数类型	描述
start_partitionId	否	String	从该分区值开始返回分区列表，返回的分区列表不包括此分区。 必须将“start_partitionId”的值设置为“shardId-0000000xxx”格式，xxx为要查询的通道分区Id。

参数	是否必选	参数类型	描述
limit_partitions	否	Integer	单次请求返回的最大分区数。

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）。

响应参数

状态码：200

表 5-19 响应 Body 参数

参数	参数类型	描述
stream_name	String	通道名称。
create_time	Long	通道创建的时间，13位时间戳。
last_modified_time	Long	通道最近一次修改的时间，13位时间戳。
status	String	通道的当前状态。 • CREATING：创建中 • RUNNING：运行中 • TERMINATING：删除中 • TERMINATED：已删除
stream_type	String	通道类型。 • COMMON：普通通道，表示1MB带宽。 • ADVANCED：高级通道，表示5MB带宽。
partitions	Array of **PartitionResult** objects	通道的分区列表。

参数	参数类型	描述
has_more_partitions	Boolean	是否还有更多满足请求条件的分区。 ● 是：true。 ● 否：false。
retention_period	Integer	数据保留时长，单位是小时。
stream_id	String	通道唯一标示符。
data_type	String	源数据类型。 ● BLOB：存储在数据库管理系统中的一组二进制数据。 ● JSON：一种开放的文件格式，以易读的文字为基础，用来传输由属性值或者序列性的值组成的数据对象。 ● CSV：纯文本形式存储的表格数据，分隔符默认采用逗号。 缺省值：BLOB。
data_schema	String	用于描述用户JSON、CSV格式的源数据结构，采用Avro Schema的语法描述。Avro介绍您也可以点击**这里**查看。
compression_format	String	数据的压缩类型，目前支持： ● snappy ● gzip ● zip 默认不压缩。
csv_properties	**CSVProperties** object	CSV 格式数据的相关属性，比如分隔符 delimiter
writable_partition_count	Integer	可写分区总数量（只包含ACTIVE状态的分区）。
readable_partition_count	Integer	可读分区总数量（包含ACTIVE与DELETED状态的分区）。
update_partition_counts	Array of **UpdatePartitionCount** objects	扩缩容操作记录列表。
tags	Array of **Tag** objects	通道的标签列表。
sys_tags	Array of **SysTag** objects	通道的企业项目。

参数	参数类型	描述
auto_scale_enabled	Boolean	是否开启自动扩缩容。 ● true：开启自动扩缩容。 ● false：关闭自动扩缩容。 默认不开启。
auto_scale_min_partition_count	Integer	当自动扩缩容启用时，自动缩容的最小分片数。
auto_scale_max_partition_count	Integer	当自动扩缩容启用时，自动扩容的最大分片数。

表 5-20 PartitionResult

参数	参数类型	描述
status	String	分区的当前状态。 ● CREATING：创建中 ● ACTIVE：可用 ● DELETED：删除中 ● EXPIRED：已过期
partition_id	String	分区的唯一标识符。
hash_range	String	分区的可能哈希键值范围。
sequence_number_range	String	分区的序列号范围。
parent_partitions	String	父分区。

表 5-21 CSVProperties

参数	参数类型	描述
delimiter	String	数据分隔符。

表 5-22 UpdatePartitionCount

参数	参数类型	描述
create_timestamp	Long	扩缩容操作执行时间戳，13位时间戳。
src_partition_count	Integer	扩缩容操作前分区数量。

参数	参数类型	描述
target_partition_count	Integer	扩缩容操作后分区数量。
result_code	Integer	扩缩容操作响应码。
result_msg	Integer	扩缩容操作响应信息。
auto_scale	Boolean	本次扩缩容操作是否为自动扩缩容。 ● true：自动扩缩容。 ● false：手动扩缩容。

表 5-23 Tag

参数	参数类型	描述
key	String	键。 ● 不能为空。 ● 对于同一资源键值唯一。 ● 字符集：A-Z，a-z ， 0-9，‘-’，‘_’，UNICODE字符（\u4E00-\u9FFF）。
value	String	值。 ● 长度不超过43个字符。 ● 字符集：A-Z，a-z ， 0-9，‘.’，‘-’，‘_’，UNICODE字符（\u4E00-\u9FFF）。 ● 只能包含数字、字母、中划线“-”、下划线“_”。

表 5-24 SysTag

参数	参数类型	描述
key	String	键。 ● 不能为空。 ● 值必须为_sys_enterprise_project_id。
value	String	值。 ● 对应的是企业项目ID，需要在企业管理页面获取。 ● 36位UUID。

请求示例

查看通道详情

```
GET https://{Endpoint}/v2/{project_id}/streams/{stream_name}
```

响应示例

状态码：200

正常返回

```
{
  "stream_id" : "8QM3Nt9YTLOwtUVYJhO",
  "stream_name" : "newstream",
  "create_time" : 1593569685875,
  "last_modified_time" : "1599050091026,",
  "retention_period" : 24,
  "status" : "RUNNING",
  "stream_type" : "COMMON",
  "data_type" : "BLOB",
  "writable_partition_count" : 1,
  "readable_partition_count" : 1,
  "tags" : [ ],
  "auto_scale_enabled" : false,
  "auto_scale_min_partition_count" : 0,
  "auto_scale_max_partition_count" : 0,
  "partitions" : [ {
    "status" : "ACTIVE",
    "partition_id" : "shardId-0000000000",
    "hash_range" : "[0 : 92233720368547758O7]",
    "sequence_number_range" : "[289911 : 289927]"
  } ],
  "has_more_partitions" : false
}
```

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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class ShowStreamSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowStreamRequest request = new ShowStreamRequest();
        request.withStreamName("{stream_name}");
        try {
            ShowStreamResponse response = client.showStream(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 huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowStreamRequest()
        request.stream_name = "{stream_name}"
        response = client.show_stream(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"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowStreamRequest{}
    request.StreamName = "{stream_name}"
    response, err := client.ShowStream(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	正常返回

错误码

请参见**错误码**。

5.1.5 修改分区数量

功能介绍

本接口用于变更指定通道中的分区数量。

调用方法

请参见**如何调用API**。

URI

PUT /v2/{project_id}/streams/{stream_name}

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。
stream_name	是	String	需要变更分区数量的通道名称。

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。 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 5-27 请求 Body 参数

参数	是否必选	参数类型	描述
stream_name	是	String	需要变更分区数量的通道名称。

参数	是否必选	参数类型	描述
target_partition_count	是	Integer	变更的目标分区数量。 取值为大于0的整数。 设置的值大于当前分区数量表示扩容，小于当前分区数量表示缩容。 注意： 每个通道在一小时内扩容和缩容总次数最多5次，且一小时内扩容或缩容操作有一次成功则最近一小时内不允许再次进行扩容或缩容操作。

响应参数

无

请求示例

变更通道分区数量

```
PUT https://{Endpoint}/v2/{project_id}/streams/{stream_name}

{
  "stream_name" : "newstream",
  "target_partition_count" : 5
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

变更通道分区数量

```
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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class UpdatePartitionCountSolution {
```

```
    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        UpdatePartitionCountRequest request = new UpdatePartitionCountRequest();
        request.withStreamName("{stream_name}");
        UpdatePartitionCountReq body = new UpdatePartitionCountReq();
        body.withTargetPartitionCount(5);
        body.withStreamName("newstream");
        request.withBody(body);
        try {
            UpdatePartitionCountResponse response = client.updatePartitionCount(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

变更通道分区数量

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
```

```
        .build()

    try:
        request = UpdatePartitionCountRequest()
        request.stream_name = "{stream_name}"
        request.body = UpdatePartitionCountReq(
            target_partition_count=5,
            stream_name="newstream"
        )
        response = client.update_partition_count(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

变更通道分区数量

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdatePartitionCountRequest{}
    request.StreamName = "{stream_name}"
    request.Body = &model.UpdatePartitionCountReq{
        TargetPartitionCount: int32(5),
        StreamName: "newstream",
    }
    response, err := client.UpdatePartitionCount(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	正常返回

错误码

请参见**错误码**。

5.1.6 更新通道信息

功能介绍

本接口用于更新指定通道的通道信息。

调用方法

请参见**如何调用API**。

URI

PUT /v3/{project_id}/streams/{stream_name}

表 5-28 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
stream_name	是	String	需要变更分区数量的通道名称。

请求参数

表 5-29 请求 Body 参数

参数	是否必选	参数类型	描述
stream_name	是	String	待更新的通道名称。

参数	是否必选	参数类型	描述
data_duration	否	Integer	数据保留时长。 取值范围：24~72。 单位：小时。 空表示使用缺省值。
data_type	否	String	源数据类型。 ● BLOB：存储在数据库管理系统中的一组二进制数据。 ● JSON：一种开放的文件格式，以易读的文字为基础，用来传输由属性值或者序列性的值组成的数据对象。 ● CSV：纯文本形式存储的表格数据，分隔符默认采用逗号。 缺省值：BLOB。
data_schema	否	String	用于描述用户JSON、CSV格式的源数据结构，采用Avro Schema的语法描述。
auto_scale_enabled	否	Boolean	是否开启自动扩缩容。 ● true：开启自动扩缩容。 ● false：关闭自动扩缩容。 默认不开启。
auto_scale_min_partition_count	否	Long	当自动扩缩容启用时，自动缩容的最小分片数。
auto_scale_max_partition_count	否	Long	当自动扩缩容启用时，自动扩容的最大分片数。

响应参数

无

请求示例

- 更新通道生命周期

```
PUT https://{Endpoint}/v3/{project_id}/streams/{stream_name}

{
  "stream_name" : "stz_test",
  "data_duration" : 48
}
```

- 更新通道类型

```
PUT https://{Endpoint}/v3/{project_id}/streams/{stream_name}

{
  "stream_name" : "stz_test",
  "data_type" : "JSON"
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

- 更新通道生命周期

```
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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class UpdateStreamSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        UpdateStreamRequest request = new UpdateStreamRequest();
        request.withStreamName("{stream_name}");
        UpdateStreamReq body = new UpdateStreamReq();
        body.withDataDuration(48);
        body.withStreamName("stz_test");
        request.withBody(body);
        try {
            UpdateStreamResponse response = client.updateStream(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());
        }
    }
}
```

- 更新通道类型

```
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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class UpdateStreamSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        UpdateStreamRequest request = new UpdateStreamRequest();
        request.withStreamName("{stream_name}");
        UpdateStreamReq body = new UpdateStreamReq();
        body.withDataType(UpdateStreamReq.DataTypeEnum.fromValue("JSON"));
        body.withStreamName("stz_test");
        request.withBody(body);
        try {
            UpdateStreamResponse response = client.updateStream(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

- 更新通道生命周期

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UpdateStreamRequest()
        request.stream_name = "{stream_name}"
        request.body = UpdateStreamReq(
            data_duration=48,
            stream_name="stz_test"
        )
        response = client.update_stream(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

- 更新通道类型

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
```

```
        .build()

    try:
        request = UpdateStreamRequest()
        request.stream_name = "{stream_name}"
        request.body = UpdateStreamReq(
            data_type="JSON",
            stream_name="stz_test"
        )
        response = client.update_stream(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

- 更新通道生命周期

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdateStreamRequest{}
    request.StreamName = "{stream_name}"
    dataDurationUpdateStreamReq:= int32(48)
    request.Body = &model.UpdateStreamReq{
        DataDuration: &dataDurationUpdateStreamReq,
        StreamName: "stz_test",
    }
    response, err := client.UpdateStream(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

- 更新通道类型

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdateStreamRequest{}
    request.StreamName = "{stream_name}"
    dataTypeUpdateStreamReq:= model.GetUpdateStreamReqDataTypeEnum().JSON
    request.Body = &model.UpdateStreamReq{
        DataType: &dataTypeUpdateStreamReq,
        StreamName: "stz_test",
    }
    response, err := client.UpdateStream(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
204	正常返回

错误码

请参见**错误码**。

5.1.7 添加权限策略

功能介绍

本接口用于给指定通道添加权限策略。

调用方法

请参见**如何调用API**。

URI

POST /v2/{project_id}/streams/{stream_name}/policies

表 5-30 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
stream_name	是	String	需要添加授权策略的通道名称。

请求参数

表 5-31 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 5-32 请求 Body 参数

参数	是否必选	参数类型	描述
stream_id	是	String	通道唯一标识符。
principal_name	是	String	授权用户。 如果授权给指定租户，格式为：domainName.*；如果授权给租户下的指定子用户，则格式为：domainName.userName； 支持多账号添加，用","隔开，比如：domainName1.userName1,domainName2.userName2；

参数	是否必选	参数类型	描述
action_type	是	String	授权操作类型。 • putRecords：上传数据。 • getRecords：下载数据。 • getStreamInfo：通道详情。
effect	是	String	授权影响类型。 • accept：允许该授权操作。

响应参数

无

请求示例

- 给租户添加权限策略

```
POST https://{Endpoint}/v2/{project_id}/streams/{stream_name}/policies

{
  "stream_id" : "CiFdELMr0401K9GGZlp",
  "principal_name" : "domainname1",
  "action_type" : "putRecords",
  "effect" : "accept"
}
```

- 给子用户添加权限策略

```
POST https://{Endpoint}/v2/{project_id}/streams/{stream_name}/policies

{
  "stream_id" : "CiFdELMr0401K9GGZlp",
  "principal_name" : "domainname1.username1",
  "action_type" : "putRecords",
  "effect" : "accept"
}
```

响应示例

无

状态码

状态码	描述
200	正常返回

错误码

请参见**错误码**。

5.1.8 查询权限策略列表

功能介绍

本接口用于查询指定通道的权限策略列表。

调用方法

请参见**如何调用API**。

URI

GET /v2/{project_id}/streams/{stream_name}/policies

表 5-33 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
stream_name	是	String	通道名称。

请求参数

表 5-34 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 5-35 响应 Body 参数

参数	参数类型	描述
stream_id	String	通道唯一标识符。
rules	Array of **PrincipalRule** objects	通道授权信息列表。

表 5-36 PrincipalRule

参数	参数类型	描述
principal	String	授权用户ID。
principal_name	String	授权用户名。如果授权给租户下的所有子用户，格式为：domainName.*；如果授权给租户下的指定子用户，则格式为：domainName.userName
action_type	String	授权操作类型。 ● putRecords：上传数据。 ● getRecords：下载数据。
effect	String	授权影响类型。 ● accept：允许该授权操作。

请求示例

查询权限策略列表

```
GET https://{Endpoint}/v2/{project_id}/streams/{stream_name}/policies
```

响应示例

状态码：200

正常返回

```
{
  "streamId" : "CiFdELMr0401K9GGZlp",
  "rules" : [ {
    "action_type" : "putRecords",
    "principal" : "3b3f237122574xxxxb74482ae11005ba.*",
    "principal_name" : "anotherusername",
    "effect" : "accept"
  } ]
}
```

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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;
```

```
public class ListPoliciesSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        ListPoliciesRequest request = new ListPoliciesRequest();
        request.withStreamName("{stream_name}");
        try {
            ListPoliciesResponse response = client.listPolicies(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 huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListPoliciesRequest()
```

```
        request.stream_name = "{stream_name}"
        response = client.list_policies(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"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListPoliciesRequest{}
    request.StreamName = "{stream_name}"
    response, err := client.ListPolicies(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	正常返回

错误码

请参见**错误码**。

5.2 App 管理

5.2.1 创建消费 App

功能介绍

本接口用于创建消费APP。

调用方法

请参见**如何调用API**。

URI

POST /v2/{project_id}/apps

表 5-37 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

请求参数

表 5-38 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 5-39 请求 Body 参数

参数	是否必选	参数类型	描述
app_name	是	String	APP的名称，用户数据消费程序的唯一标识符。 应用名称由字母、数字、下划线和中划线组成，长度为1～200个字符。

响应参数

无

请求示例

创建消费App

```
POST https://{Endpoint}/v2/{project_id}/apps

{
  "app_name" : "newapp"
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

创建消费App

```
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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class CreateAppSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateAppRequest request = new CreateAppRequest();
        CreateAppReq body = new CreateAppReq();
        body.withAppName("newapp");
        request.withBody(body);
        try {
```

```
            CreateAppResponse response = client.createApp(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

创建消费App

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateAppRequest()
        request.body = CreateAppReq(
            app_name="newapp"
        )
        response = client.create_app(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

创建消费App

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
```

```
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateAppRequest{}
    request.Body = &model.CreateAppReq{
        AppName: "newapp",
    }
    response, err := client.CreateApp(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
201	正常返回

错误码

请参见**错误码**。

5.2.2 查询 App 列表

功能介绍

本接口用于查询APP列表。

调用方法

请参见**如何调用API**。

URI

GET /v2/{project_id}/apps

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。

表 5-41 Query 参数

参数	是否必选	参数类型	描述
limit	否	Integer	单次请求返回APP列表的最大数量。
start_app_name	否	String	从该app名称开始返回app列表，返回的app列表不包括此app名称。
stream_name	否	String	返回该通道下的app列表。

请求参数

表 5-42 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 5-43 响应 Body 参数

参数	参数类型	描述
has_more_app	Boolean	是否还有更多满足条件的App。 • true：是。 • false：否。
apps	Array of DescribeAppResult objects	AppEntry list that meets the current request.
total_number	Integer	满足条件的App总数。

表 5-44 DescribeAppResult

参数	参数类型	描述
app_name	String	App的名称。
app_id	String	App的唯一标识符。
create_time	Long	App创建的时间，单位毫秒。
commit_checkpoint_stream_names	Array of strings	关联通道列表。

请求示例

查询App列表

```
GET https://{Endpoint}/v2/{project_id}/apps
```

响应示例

状态码：200

正常返回

```
{
  "total_number" : 1,
  "apps" : [ {
    "app_id" : "bd6IPpvgiIflQPMpi9M",
    "app_name" : "newstream",
    "create_time" : 1593569685875
  } ],
  "has_more_app" : true
}
```

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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class ListAppSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        ListAppRequest request = new ListAppRequest();
        try {
            ListAppResponse response = client.listApp(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 huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListAppRequest()
        response = client.list_app(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"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListAppRequest{}
    response, err := client.ListApp(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	正常返回
400	Invalid Parameters
404	Application not found
500	Internal Server Error

错误码

请参见**错误码**。

5.2.3 删除 App

功能介绍

本接口用于删除App。

调用方法

请参见**如何调用API**。

URI

DELETE /v2/{project_id}/apps/{app_name}

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。
app_name	是	String	需要删除的App名称。

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。 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

无

请求示例

删除App

```
DELETE https://{Endpoint}/v2/{project_id}/apps/{app_name}
```

响应示例

无

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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class DeleteAppSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        DeleteAppRequest request = new DeleteAppRequest();
        request.withAppName("{app_name}");
        try {
            DeleteAppResponse response = client.deleteApp(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 huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteAppRequest()
        request.app_name = "{app_name}"
        response = client.delete_app(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"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteAppRequest{}
    request.AppName = "{app_name}"
    response, err := client.DeleteApp(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
204	正常返回

错误码

请参见**错误码**。

5.2.4 查看 App 详情

功能介绍

本接口用于查询APP详情。

调用方法

请参见**如何调用API**。

URI

GET /v2/{project_id}/apps/{app_name}

表 5-47 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

参数	是否必选	参数类型	描述
app_name	是	String	需要查询的App名称。

请求参数

表 5-48 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 5-49 响应 Body 参数

参数	参数类型	描述
app_name	String	App的名称。
app_id	String	App的唯一标识符。
create_time	Long	App创建的时间，单位毫秒。
commit_checkpoint_stream_names	Array of strings	关联通道列表。

请求示例

查询App详情

```
GET https://{Endpoint}/v2/{project_id}/apps/{app_name}
```

响应示例

状态码：200

正常返回

```
{
  "app_id" : "bd6IPpvgiIflQPMpi9M",
  "app_name" : "newstream",
  "create_time" : 1593569685875,
  "commit_checkpoint_stream_names" : [ "newstream" ]
}
```

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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class ShowAppSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowAppRequest request = new ShowAppRequest();
        request.withAppName("{app_name}");
        try {
            ShowAppResponse response = client.showApp(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 huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowAppRequest()
        request.app_name = "{app_name}"
        response = client.show_app(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"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowAppRequest{}
    request.AppName = "{app_name}"
    response, err := client.ShowApp(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	正常返回

错误码

请参见**错误码**。

5.2.5 查看 App 消费状态

功能介绍

本接口用于查询APP消费状态。

调用方法

请参见**如何调用API**。

URI

GET /v2/{project_id}/apps/{app_name}/streams/{stream_name}

表 5-50 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
app_name	是	String	需要查询的App名称。
stream_name	是	String	需要查询的通道名称。

表 5-51 Query 参数

参数	是否必选	参数类型	描述
limit	否	Integer	单次请求返回的最大分区数。最小值是1，最大值是1000；默认值是100。

参数	是否必选	参数类型	描述
start_partition_id	否	String	从该分区值开始返回分区列表，返回的分区列表不包括此分区。
checkpoint_type	是	String	Checkpoint类型。 • LAST_READ：在数据库中只记录序列号。

请求参数

表 5-52 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 5-53 响应 Body 参数

参数	参数类型	描述
has_more	Boolean	是否还有更多满足条件的App。 • true：是。 • false：否。
stream_name	String	要查询的通道名称。
app_name	String	要查询的APP的名称，用户数据消费程序的唯一标识符。
partition_consuming_states	Array of **PartitionConsumingStates** objects	当前分区消费状态.

表 5-54 PartitionConsumingStates

参数	参数类型	描述
partition_id	String	通道的分区标识符。可定义为如下两种样式：- shardId-0000000000- 0比如一个通道有三个分区，那么分区标识符分别为0, 1, 2，或者shardId-0000000000, shardId-0000000001, shardId-0000000002
sequence_number	String	需要提交的序列号，用来记录该通道的消费检查点，需要保证该序列号处于有效范围内。
latest_offset	Long	索引位置, 最新的一条索引位置。
earliest_offset	Long	索引位置, 最早的一条索引位置。
checkpoint_type	String	Checkpoint类型。 ● LAST_READ：在数据库中只记录序列号。
status	String	分区的当前状态。 ● CREATING：创建中 ● ACTIVE：可用 ● DELETED：删除中 ● EXPIRED：已过期

请求示例

查询App消费状态

```
GET https://{Endpoint}/v2/{project_id}/apps/{app_name}/streams/{stream_name}
```

响应示例

状态码：200

正常返回

```
{
  "has_more" : false,
  "stream_name" : "disMonitorTest",
  "app_name" : "4aSiZ",
  "partition_consuming_states" : [ {
    "partition_id" : 0,
    "sequence_number" : -1,
    "latest_offset" : 1658297359,
    "earliest_offset" : 1653120573,
    "checkpoint_type" : "LAST_READ",
    "status" : "ACTIVE"
  } ]
}
```

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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class ShowConsumerStateSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowConsumerStateRequest request = new ShowConsumerStateRequest();
        request.withAppName("{app_name}");
        request.withStreamName("{stream_name}");
        try {
            ShowConsumerStateResponse response = client.showConsumerState(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 huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowConsumerStateRequest()
        request.app_name = "{app_name}"
        request.stream_name = "{stream_name}"
        response = client.show_consumer_state(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"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowConsumerStateRequest{}
    request.AppName = "{app_name}"
    request.StreamName = "{stream_name}"
    response, err := client.ShowConsumerState(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	正常返回

错误码

请参见**错误码**。

5.3 Checkpoint 管理

5.3.1 提交 Checkpoint

功能介绍

本接口用于提交Checkpoint。

调用方法

请参见**如何调用API**。

URI

POST /v2/{project_id}/checkpoints

表 5-55 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

请求参数

表 5-56 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 5-57 请求 Body 参数

参数	是否必选	参数类型	描述
app_name	是	String	APP的名称，用户数据消费程序的唯一标识符，需要先通过创建App接口创建。
checkpoint_type	是	String	Checkpoint类型。 ● LAST_READ：在数据库中只记录序列号。
stream_name	是	String	已创建的通道名称。
partition_id	是	String	通道的分区标识符。可定义为如下两种样式：- shardId-0000000000- 0比如一个通道有三个分区，那么分区标识符分别为0, 1, 2，或者shardId-0000000000, shardId-0000000001, shardId-0000000002
sequence_number	是	String	需要提交的序列号，用来记录该通道的消费检查点，需要保证该序列号处于有效范围内。
metadata	否	String	用户消费程序端的元数据信息。 元数据信息的最大长度为1000个字符。

响应参数

无

请求示例

提交Checkpoint

```
POST https://{Endpoint}/v2/{project_id}/checkpoints
```

```
{
  "stream_name" : "newstream",
  "app_name" : "newapp",
  "partition_id" : "0",
  "sequence_number" : "2",
  "checkpoint_type" : "LAST_READ"
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

提交Checkpoint

```
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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class CommitCheckpointSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        CommitCheckpointRequest request = new CommitCheckpointRequest();
        CommitCheckpointReq body = new CommitCheckpointReq();
        body.withSequenceNumber("2");
        body.withPartitionId("0");
        body.withStreamName("newstream");
        body.withCheckpointType(CommitCheckpointReq.CheckpointTypeEnum.fromValue("LAST_READ"));
        body.withAppName("newapp");
        request.withBody(body);
        try {
            CommitCheckpointResponse response = client.commitCheckpoint(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

提交Checkpoint

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CommitCheckpointRequest()
        request.body = CommitCheckpointReq(
            sequence_number="2",
            partition_id="0",
            stream_name="newstream",
            checkpoint_type="LAST_READ",
            app_name="newapp"
        )
        response = client.commit_checkpoint(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

提交Checkpoint

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
```

```
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CommitCheckpointRequest{}
    request.Body = &model.CommitCheckpointReq{
        SequenceNumber: "2",
        PartitionId: "0",
        StreamName: "newstream",
        CheckpointType: model.GetCommitCheckpointReqCheckpointTypeEnum().LAST_READ,
        AppName: "newapp",
    }
    response, err := client.CommitCheckpoint(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
201	正常返回

错误码

请参见**错误码**。

5.3.2 查询 Checkpoint 详情

功能介绍

本接口用于查询Checkpoint详情。

调用方法

请参见**如何调用API**。

URI

GET /v2/{project_id}/checkpoints

表 5-58 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

表 5-59 Query 参数

参数	是否必选	参数类型	描述
stream_name	是	String	该Checkpoint所属的通道名称。
partition_id	是	String	该Checkpoint所属的通道分区标识符。可定义为如下两种样式：- shardId-0000000000- 0比如一个通道有三个分区，那么分区标识符分别为0, 1, 2，或者shardId-0000000000, shardId-0000000001, shardId-0000000002
app_name	是	String	该Checkpoint关联App名称。
checkpoint_type	是	String	Checkpoint类型。 • LAST_READ：在数据库中只记录序列号。

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）。

响应参数

状态码：200

表 5-61 响应 Body 参数

参数	参数类型	描述
sequence_number	String	序列号，用来记录该通道的消费检查点。
metadata	String	用户消费程序端的元数据信息。

请求示例

查询Checkpoint详情

```
GET https://{Endpoint}/v2/{project_id}/checkpoints
```

响应示例

状态码：200

正常返回

```
{
  "sequence_number" : "newstram",
  "metadata" : ""
}
```

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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class ShowCheckpointSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowCheckpointRequest request = new ShowCheckpointRequest();
        try {
            ShowCheckpointResponse response = client.showCheckpoint(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 huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
```

```
        .build()

    try:
        request = ShowCheckpointRequest()
        response = client.show_checkpoint(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"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowCheckpointRequest{}
    response, err := client.ShowCheckpoint(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	正常返回

错误码

请参见**错误码**。

5.3.3 删除 Checkpoint

功能介绍

本接口用于删除Checkpoint。

调用方法

请参见**如何调用API**。

URI

DELETE /v2/{project_id}/checkpoints

表 5-62 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

表 5-63 Query 参数

参数	是否必选	参数类型	描述
stream_name	是	String	该Checkpoint所属的通道名称。
app_name	是	String	该Checkpoint关联App名称。
checkpoint_type	是	String	Checkpoint类型。 LAST_READ：在数据库中只记录序列号。
partition_id	否	String	该Checkpoint所属的通道分区标识符。可定义为如下两种样式：- shardId-0000000000- 0比如一个通道有三个分区，那么分区标识符分别为0, 1, 2，或者shardId-0000000000, shardId-0000000001, shardId-0000000002

请求参数

表 5-64 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

无

请求示例

删除Checkpoint

```
DELETE https://{Endpoint}/v2/{project_id}/checkpoints
```

响应示例

无

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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class DeleteCheckpointSolution {

    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);

    DisClient client = DisClient.newBuilder()
            .withCredential(auth)
            .withRegion(DisRegion.valueOf("<YOUR REGION>"))
            .build();
    DeleteCheckpointRequest request = new DeleteCheckpointRequest();
    try {
        DeleteCheckpointResponse response = client.deleteCheckpoint(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 huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteCheckpointRequest()
        response = client.delete_checkpoint(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"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
```

```
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteCheckpointRequest{}
    response, err := client.DeleteCheckpoint(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
204	正常返回

错误码

请参见**错误码**。

5.4 数据管理

5.4.1 上传数据

功能介绍

本接口用于上传数据到DIS通道中。

调用方法

请参见**如何调用API**。

URI

POST /v2/{project_id}/records

表 5-65 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

请求参数

表 5-66 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 5-67 请求 Body 参数

参数	是否必选	参数类型	描述
stream_name	是	String	已创建的通道名称。
stream_id	否	String	通道唯一标识符。 当使用stream_name没有找到对应通道且stream_id不为空时，会使用stream_id去查找通道。 说明： 上传数据到被授权的通道时，必须配置此参数。
records	是	Array of **PutRecordsRequestEntry** objects	待上传的记录列表。

表 5-68 PutRecordsRequestEntry

参数	是否必选	参数类型	描述
data	是	String	需要上传的数据。 上传的数据为序列化之后的二进制数据（Base64编码后的字符串）。 比如需要上传字符串“data”，“data”经过Base64编码之后是“ZGF0YQ==”。
explicit_hash_key	否	String	用于明确数据需要写入分区的哈希值，此哈希值将覆盖“partition_key”的哈希值。 取值范围：0~long.max
partition_id	否	String	通道的分区标识符。可定义为如下两种样式：- shardId-0000000000- 0比如一个通道有三个分区，那么分区标识符分别为0, 1, 2，或者shardId-0000000000, shardId-0000000001, shardId-0000000002
partition_key	否	String	数据将写入的分区。说明：如果传了partition_id参数，则优先使用partition_id参数。如果partition_id没有传，则使用partition_key。

响应参数

状态码：200

表 5-69 响应 Body 参数

参数	参数类型	描述
failed_record_count	Integer	上传失败的数据数量。
records	Array of **PutRecordsResultEntry** objects	上传结果列表。

表 5-70 PutRecordsResultEntry

参数	参数类型	描述
partition_id	String	数据上传到的分区ID。
sequence_number	String	数据上传到的序列号。序列号是每个记录的唯一标识符。序列号由DIS在数据生产者调用PutRecords操作以添加数据到DIS数据通道时DIS服务自动分配的。同一分区键的序列号通常会随时间变化增加。PutRecords请求之间的时间段越长，序列号越大。
error_code	String	错误码。
error_message	String	错误消息。

请求示例

上传数据

```
POST https://{Endpoint}/v2/{project_id}/records

{
  "stream_name" : "newstream",
  "records" : [ {
    "data" : "MTExMTExMTExMTExMTExMTExMTExMTExMTExMTExMTE="
  } ]
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

上传数据

```
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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;

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

public class SendRecordsSolution {
```

```
    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        SendRecordsRequest request = new SendRecordsRequest();
        PutRecordsRequest body = new PutRecordsRequest();
        List<PutRecordsRequestEntry> listbodyRecords = new ArrayList<>();
        listbodyRecords.add(
            new PutRecordsRequestEntry()
                .withData("MTExMTExMTExMTExMTExMTExMTExMTExMTExMTExMTE=")
        );
        body.withRecords(listbodyRecords);
        body.withStreamName("newstream");
        request.withBody(body);
        try {
            SendRecordsResponse response = client.sendRecords(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

上传数据

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
    .with_credentials(credentials) \
    .with_region(DisRegion.value_of("<YOUR REGION>")) \
    .build()

try:
    request = SendRecordsRequest()
    listRecordsbody = [
        PutRecordsRequestEntry(
            data="MTExMTExMTExMTExMTExMTExMTExMTExMTExMTExMTE="
        )
    ]
    request.body = PutRecordsRequest(
        records=listRecordsbody,
        stream_name="newstream"
    )
    response = client.send_records(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

上传数据

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.SendRecordsRequest{}
    var listRecordsbody = []model.PutRecordsRequestEntry{
        {
            Data: "MTExMTExMTExMTExMTExMTExMTExMTExMTExMTExMTE=",
        },
    }
    request.Body = &model.PutRecordsRequest{
        Records: listRecordsbody,
```

```
        StreamName: "newstream",
    }
    response, err := client.SendRecords(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	正常返回

错误码

请参见**错误码**。

5.4.2 下载数据

功能介绍

本接口用于从DIS通道中下载数据。

调用方法

请参见**如何调用API**。

URI

GET /v2/{project_id}/records

表 5-71 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

表 5-72 Query 参数

参数	是否必选	参数类型	描述
partition-cursor	是	String	数据游标，需要先通过获取数据游标的接口获取。 取值范围：1~512个字符。 说明： 数据游标有效期为5分钟。
max_fetch_bytes	否	Integer	每个请求获取记录的最大字节数。 注意： 该值如果小于分区中单条记录的大小，会导致一直无法获取到记录。

请求参数

表 5-73 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 5-74 响应 Body 参数

参数	参数类型	描述
records	Array of **Record** objects	下载的记录列表。
next_partition_cursor	String	下一个迭代器。 说明： 数据游标有效期为5分钟。

表 5-75 Record

参数	参数类型	描述
partition_key	String	用户上传数据时设置的partition_key。 说明：上传数据时，如果传了partition_key参数，则下载数据时可返回此参数。如果上传数据时，未传partition_key参数，而是传入partition_id，则不返回partition_key。
sequence_number	String	该条数据的序列号。
data	String	下载的数据。 下载的数据为序列化之后的二进制数据（Base64编码后的字符串）。 比如下载数据接口返回的数据是“ZGF0YQ==”，“ZGF0YQ==”经过Base64解码之后是“data”。
timestamp	Long	记录写入DIS的时间戳。
timestamp_type	String	时间戳类型。 ● CreateTime：创建时间。

请求示例

下载数据

```
GET https://{Endpoint}/v2/{project_id}/records
```

响应示例

状态码：200

正常返回

```
{
  "records" : [ {
    "partition_key" : "0",
    "sequence_number" : "485",
    "data" : "MTExMTExMTExMTExMTExMTExMTExMTExMTExMTExMTEx",
    "timestamp" : 1527577402541,
    "timestamp_type" : "CreateTime"
  } ],
  "next_partition_cursor" :
"eyJpdGVyR2VuVGltZSI6MTQ5MDk1MDE1Nzc0NywiU3RyZWFtTmFtZSI6IjY2MCIsIlNoYXJkSWQiOiIwIiwiU2hhcmRJdGVyYXRvclR5cGUiOiJBVF9TRVFVRU5DRV9OVU1CRVIiLCJTdGFydGluZ1NlcXVlbmNlTnVtYmVyIjoiMjIiLCJUaW1lU3RhbXAiOjB9"
}
```

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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class ConsumeRecordsSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        ConsumeRecordsRequest request = new ConsumeRecordsRequest();
        try {
            ConsumeRecordsResponse response = client.consumeRecords(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 huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ConsumeRecordsRequest()
        response = client.consume_records(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"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ConsumeRecordsRequest{}
    response, err := client.ConsumeRecords(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	正常返回

错误码

请参见**错误码**。

5.4.3 获取数据游标

功能介绍

本接口用于获取数据游标。

调用方法

请参见**如何调用API**。

URI

GET /v2/{project_id}/cursors

表 5-76 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

表 5-77 Query 参数

参数	是否必选	参数类型	描述
stream-name	是	String	已创建的通道名称。
partition-id	是	String	通道的分区标识符。可定义为如下两种样式：- shardId-0000000000- 0比如一个通道有三个分区，那么分区标识符分别为0, 1, 2，或者 shardId-0000000000, shardId-0000000001, shardId-0000000002

参数	是否必选	参数类型	描述
cursor-type	否	String	游标类型。- AT_SEQUENCE_NUMBER：从特定序列号（即starting-sequence-number定义的序列号）所在的记录开始读取数据。此类型为默认游标类型。- AFTER_SEQUENCE_NUMBER：从特定序列号（即starting-sequence-number定义的序列号）后的记录开始读取数据。- TRIM_HORIZON：从最早被存储至分区的有效记录开始读取。例如，某租户使用DIS的通道，分别上传了三条数据A1，A2，A3。N天后（设定A1已过期，A2和A3仍在有效期范围内），该租户需要下载此三条数据，并选择了TRIM_HORIZON这种下载方式。那么用户可下载的数据将从A2开始读取。- LATEST：从分区中的最新记录开始读取，此设置可以保证你总是读到分区中最新记录。- AT_TIMESTAMP：从特定时间戳（即timestamp定义的时间戳）开始读取。
starting-sequence-number	否	String	序列号。序列号是每个记录的唯一标识符。序列号由DIS在数据生产者调用PutRecords操作以添加数据到DIS数据通道时DIS服务自动分配的。同一分区键的序列号通常会随时间变化增加。PutRecords请求之间的时间段越长，序列号越大。序列号与游标类型AT_SEQUENCE_NUMBER和AFTER_SEQUENCE_NUMBER强相关，二者共同确定读取数据的位置。取值范围：0~9223372036854775807。
timestamp	否	Long	开始读取数据记录的时间戳，与游标类型AT_TIMESTAMP强相关，二者共同确定读取数据的位置。 说明： 此时间戳精确到毫秒。

请求参数

表 5-78 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 5-79 响应 Body 参数

参数	参数类型	描述
partition_cursor	String	数据游标。 取值范围：1~512个字符。 说明： 数据游标有效期为5分钟。

请求示例

获取数据游标

```
GET https://{Endpoint}/v2/{project_id}/cursors
```

响应示例

状态码：200

正常返回

```
{
  "partition_cursor" :
"eyJnZXRJdGVyYXRvclBhcmFtIjp7InN0cmVhbS1uYW1lIjoianpjIiwicGFydGl0aW9uLWlkIjoiMCIsImN1cnNvci10eXBlIjoiQVRfU0VRVUVOQ0VfTlVNQkVSIiwic3RhcnRpbmctc2VxdWVuY2UtbnVtYmVyIjoiMTAifSwiZ2VuZXJhdGVUaW1lc3RhbXAiOjE1MDYxNTk1NjM0MDV9"
}
```

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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class ShowCursorSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowCursorRequest request = new ShowCursorRequest();
        try {
            ShowCursorResponse response = client.showCursor(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 huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowCursorRequest()
        response = client.show_cursor(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"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowCursorRequest{}
    response, err := client.ShowCursor(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	正常返回

错误码

请参见**错误码**。

5.5 转储任务管理

5.5.1 添加 OBS 转储任务

功能介绍

本接口用于添加OBS转储任务。

调用方法

请参见**如何调用API**。

URI

POST /v2/{project_id}/streams/{stream_name}/transfer-tasks

表 5-80 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
stream_name	是	String	已创建的通道名称。

请求参数

表 5-81 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 5-82 请求 Body 参数

参数	是否必选	参数类型	描述
destination_type	是	String	转储任务类型。 • OBS：转储到OBS • MRS：转储到MRS • DLI：转储到DLI • CLOUDTABLE：转储到CloudTable • DWS：转储到DWS
obs_destination_descriptor	否	**OBSDestinationDescriptorRequest** object	转储目的地为OBS的参数列表。

表 5-83 OBSDestinationDescriptorRequest

参数	是否必选	参数类型	描述
task_name	是	String	转储任务的名称。 任务名称由英文字母、数字、中划线和下划线组成。长度为1～64个字符。
agency_name	是	String	在统一身份认证服务(IAM)中创建委托的名称，DIS需要获取IAM委托信息去访问您指定的资源。创建委托的参数设置如下：- 委托类型：云服务- 云服务：DIS- 持续时间：永久- “所属区域”为“全局服务”，“项目”为“对象存储服务”对应的“策略”包含“Tenant Administrator”。如果已经创建过委托，可以使用IAM服务提供的查询委托列表接口，获取有效可用的委托名称。取值范围：长度不超过64位，且不可配置为空。如果有在Console控制台使用转储任务，会提示自动创建委托，自动创建的委托名称为：dis_admin_agency
deliver_time_interval	是	Integer	根据用户配置的时间，周期性的将数据导入OBS，若某个时间段内无数据，则此时间段不会生成打包文件。 单位：秒

参数	是否必选	参数类型	描述
consumer_strategy	否	String	偏移量。 ● LATEST：最大偏移量，即获取最新的数据。 ● TRIM_HORIZON：最小偏移量，即读取最早的数据。
file_prefix	否	String	在OBS中存储通道文件的自定义目录，多级目录可用“/”进行分隔，不可以“/”开头。 取值范围：英文字母、数字、下划线和斜杠，最大长度为50个字符。 默认配置为空。
partition_format	否	String	将转储文件的生成时间使用“yyyy/MM/dd/HH/mm”格式生成分区字符串，用来定义写到OBS的Object文件所在的目录层次结构。- N/A：置空，不使用日期时间目录。- yyyy：年- yyyy/MM：年/- yyyy/MM/dd：年/月/日- yyyy/MM/dd/HH：年/月/日/时- yyyy/MM/dd/HH/mm：年/月/日/时/分例如：2017/11/10/14/49，目录结构就是“2017 > 11 > 10 > 14 > 49”，“2017”表示最外层文件夹。默认值：空说明：数据转储成功后，存储的目录结构为“obs_bucket_path/file_prefix/partition_format”。
obs_bucket_path	是	String	存储该通道数据的OBS桶名称。
destination_file_type	否	String	转储文件格式。 ● text：转储目标格式为TEXT，缺省值 ● parquet：转储目标格式为Parquet ● carbon：转储目标格式为Carbon 说明： “源数据类型”为“JSON”，“转储服务类型”为“OBS”时才可选择“parquet”或“carbon”格式。

参数	是否必选	参数类型	描述
processing_schema	否	ProcessingSchema object	根据源数据的时间戳和已配置的"partition_format"生成对应的转储时间目录。将源数据的时间戳使用“yyyy/MM/dd/HH/mm”格式生成分区字符串，用来定义写到OBS的Object文件所在的目录层次结构。
record_delimiter	否	String	转储文件的记录分隔符，用于分隔写入转储文件的用户数据。 取值范围： • 逗号 ","，默认值 • 分号 ";" • 竖线 "|" • 换行符 "\n"

表 5-84 ProcessingSchema

参数	是否必选	参数类型	描述
timestamp_name	是	String	源数据时间戳的属性名称。
timestamp_type	是	String	源数据时间戳的类型。 • String • Timestamp：Long类型的13位时间戳
timestamp_format	否	String	源数据时间戳的类型为String时必选，用于根据时间戳格式生成OBS的时间目录。

响应参数

无

请求示例

- 添加OBS转储任务

```
POST https://{Endpoint}/v2/{project_id}/streams/{stream_name}/transfer-tasks

{
  "destination_type" : "OBS",
  "obs_destination_descriptor" : {
    "task_name" : "newtask",
    "consumer_strategy" : "LATEST",
    "agency_name" : "dis_admin_agency",
    "destination_file_type" : "text",
```

```
  "obs_bucket_path" : "obsbucket",
  "file_prefix" : "",
  "partition_format" : "yyyy/MM/dd/HH/mm",
  "record_delimiter" : "|",
  "deliver_time_interval" : 30
 }
}
```

- 添加OBS转储任务（转储文件格式是parquet）

```
POST https://{Endpoint}/v2/{project_id}/streams/{stream_name}/transfer-tasks

{
 "destination_type" : "OBS",
 "obs_destination_descriptor" : {
  "task_name" : "newtask",
  "consumer_strategy" : "LATEST",
  "agency_name" : "dis_admin_agency",
  "destination_file_type" : "parquet",
  "obs_bucket_path" : "obsbucket",
  "file_prefix" : "",
  "partition_format" : "yyyy/MM/dd/HH/mm",
  "record_delimiter" : "|",
  "deliver_time_interval" : 30
 }
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

- 添加OBS转储任务

```
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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class CreateObsTransferTaskSolution {

    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);
```

```
        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateObsTransferTaskRequest request = new CreateObsTransferTaskRequest();
        request.withStreamName("{stream_name}");
        CreateTransferTaskReq body = new CreateTransferTaskReq();
        OBSDestinationDescriptorRequest obsDestinationDescriptorbody = new
OBSDestinationDescriptorRequest();
        obsDestinationDescriptorbody.withTaskName("newtask")
            .withObsBucketPath("obsbucket")
            .withAgencyName("dis_admin_agency")
            .withDestinationFileType(OBSDestinationDescriptorRequest.DestinationFileTypeEnum.fromValu
e("text"))
            .withRecordDelimiter("|")
            .withDeliverTimeInterval(30)
            .withFilePrefix("")
            .withPartitionFormat(OBSDestinationDescriptorRequest.PartitionFormatEnum.fromValue("yyyy
/MM/dd/HH/mm"))
            .withConsumerStrategy(OBSDestinationDescriptorRequest.ConsumerStrategyEnum.fromValue(
"LATEST"));
        body.withObsDestinationDescriptor(obsDestinationDescriptorbody);
        body.withDestinationType(CreateTransferTaskReq.DestinationTypeEnum.fromValue("OBS"));
        request.withBody(body);
        try {
            CreateObsTransferTaskResponse response = client.createObsTransferTask(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());
        }
    }
}
```

- 添加OBS转储任务（转储文件格式是parquet）

```
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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class CreateObsTransferTaskSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateObsTransferTaskRequest request = new CreateObsTransferTaskRequest();
        request.withStreamName("{stream_name}");
        CreateTransferTaskReq body = new CreateTransferTaskReq();
        OBSDestinationDescriptorRequest obsDestinationDescriptorbody = new
OBSDestinationDescriptorRequest();
        obsDestinationDescriptorbody.withTaskName("newtask")
            .withObsBucketPath("obsbucket")
            .withAgencyName("dis_admin_agency")
            .withDestinationFileType(OBSDestinationDescriptorRequest.DestinationFileTypeEnum.fromValu
e("parquet"))
            .withRecordDelimiter("|")
            .withDeliverTimeInterval(30)
            .withFilePrefix("")
            .withPartitionFormat(OBSDestinationDescriptorRequest.PartitionFormatEnum.fromValue("yyyy
/MM/dd/HH/mm"))
            .withConsumerStrategy(OBSDestinationDescriptorRequest.ConsumerStrategyEnum.fromValue(
"LATEST"));
        body.withObsDestinationDescriptor(obsDestinationDescriptorbody);
        body.withDestinationType(CreateTransferTaskReq.DestinationTypeEnum.fromValue("OBS"));
        request.withBody(body);
        try {
            CreateObsTransferTaskResponse response = client.createObsTransferTask(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

- 添加OBS转储任务

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
```

```
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateObsTransferTaskRequest()
        request.stream_name = "{stream_name}"
        obsDestinationDescriptorbody = OBSDestinationDescriptorRequest(
            task_name="newtask",
            obs_bucket_path="obsbucket",
            agency_name="dis_admin_agency",
            destination_file_type="text",
            record_delimiter="|",
            deliver_time_interval=30,
            file_prefix="",
            partition_format="yyyy/MM/dd/HH/mm",
            consumer_strategy="LATEST"
        )
        request.body = CreateTransferTaskReq(
            obs_destination_descriptor=obsDestinationDescriptorbody,
            destination_type="OBS"
        )
        response = client.create_obs_transfer_task(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

- 添加OBS转储任务（转储文件格式是parquet）

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateObsTransferTaskRequest()
        request.stream_name = "{stream_name}"
        obsDestinationDescriptorbody = OBSDestinationDescriptorRequest(
            task_name="newtask",
            obs_bucket_path="obsbucket",
            agency_name="dis_admin_agency",
            destination_file_type="parquet",
            record_delimiter="|",
            deliver_time_interval=30,
            file_prefix="",
            partition_format="yyyy/MM/dd/HH/mm",
            consumer_strategy="LATEST"
```

```
        )
        request.body = CreateTransferTaskReq(
            obs_destination_descriptor=obsDestinationDescriptorbody,
            destination_type="OBS"
        )
        response = client.create_obs_transfer_task(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

- 添加OBS转储任务

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateObsTransferTaskRequest{}
    request.StreamName = "{stream_name}"
    destinationFileTypeObsDestinationDescriptor:= model.GetObsDestinationDescriptorRequestDestinationFileTypeEnum().TEXT
    recordDelimiterObsDestinationDescriptor:= "|"
    filePrefixObsDestinationDescriptor:= ""
    partitionFormatObsDestinationDescriptor:= model.GetObsDestinationDescriptorRequestPartitionFormatEnum().YYYY_MM_DD_HH_MM
    consumerStrategyObsDestinationDescriptor:= model.GetObsDestinationDescriptorRequestConsumerStrategyEnum().LATEST
    obsDestinationDescriptorbody := &model.ObsDestinationDescriptorRequest{
        TaskName: "newtask",
        ObsBucketPath: "obsbucket",
        AgencyName: "dis_admin_agency",
        DestinationFileType: &destinationFileTypeObsDestinationDescriptor,
        RecordDelimiter: &recordDelimiterObsDestinationDescriptor,
        DeliverTimeInterval: int32(30),
        FilePrefix: &filePrefixObsDestinationDescriptor,
        PartitionFormat: &partitionFormatObsDestinationDescriptor,
        ConsumerStrategy: &consumerStrategyObsDestinationDescriptor,
```

```
    }
    request.Body = &model.CreateTransferTaskReq{
        ObsDestinationDescriptor: obsDestinationDescriptorbody,
        DestinationType: model.GetCreateTransferTaskReqDestinationTypeEnum().OBS,
    }
    response, err := client.CreateObsTransferTask(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

- 添加OBS转储任务（转储文件格式是parquet）

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateObsTransferTaskRequest{}
    request.StreamName = "{stream_name}"
    destinationFileTypeObsDestinationDescriptor:= model.GetObsDestinationDescriptorRequestDestinationFileTypeEnum().PARQUET
    recordDelimiterObsDestinationDescriptor:= "|"
    filePrefixObsDestinationDescriptor:= ""
    partitionFormatObsDestinationDescriptor:= model.GetObsDestinationDescriptorRequestPartitionFormatEnum().YYYY_MM_DD_HH_MM
    consumerStrategyObsDestinationDescriptor:= model.GetObsDestinationDescriptorRequestConsumerStrategyEnum().LATEST
    obsDestinationDescriptorbody := &model.ObsDestinationDescriptorRequest{
        TaskName: "newtask",
        ObsBucketPath: "obsbucket",
        AgencyName: "dis_admin_agency",
        DestinationFileType: &destinationFileTypeObsDestinationDescriptor,
        RecordDelimiter: &recordDelimiterObsDestinationDescriptor,
        DeliverTimeInterval: int32(30),
        FilePrefix: &filePrefixObsDestinationDescriptor,
        PartitionFormat: &partitionFormatObsDestinationDescriptor,
        ConsumerStrategy: &consumerStrategyObsDestinationDescriptor,
    }
    request.Body = &model.CreateTransferTaskReq{
        ObsDestinationDescriptor: obsDestinationDescriptorbody,
```

```
        DestinationType: model.GetCreateTransferTaskReqDestinationTypeEnum().OBS,
    }
    response, err := client.CreateObsTransferTask(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
201	正常返回

错误码

请参见**错误码**。

5.5.2 查询转储任务列表

功能介绍

本接口用于查询转储任务列表。

调用方法

请参见**如何调用API**。

URI

GET /v2/{project_id}/streams/{stream_name}/transfer-tasks

表 5-85 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
stream_name	是	String	需要查询的通道名称。

请求参数

表 5-86 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 5-87 响应 Body 参数

参数	参数类型	描述
total_number	Integer	转储任务总数。
quota	Integer	可创建的转储任务配额。
tasks	Array of **TransferTask** objects	转储任务列表。

表 5-88 TransferTask

参数	参数类型	描述
task_name	String	转储任务名称。
state	String	转储任务状态。 • ERROR：错误。 • STARTING：启动中。 • PAUSED：已停止。 • RUNNING：运行中。 • DELETE：已删除。 • ABNORMAL：异常。

参数	参数类型	描述
destination_type	String	转储任务类型。 ● OBS：转储到OBS。 ● MRS：转储到MRS。 ● DLI：转储到DLI。 ● CLOUDTABLE：转储到CloudTable。 ● DWS：转储到DWS。
create_time	Long	转储任务创建时间。
last_transfer_timestamp	Long	转储任务最近一次转储时间。

请求示例

查询转储任务列表

```
GET https://{Endpoint}/v2/{project_id}/streams/{stream_name}/transfer-tasks
```

响应示例

状态码：200

正常返回

```
{
  "tasks" : [ {
    "task_id" : "As805BudhcH1lDs6gbn",
    "destination_type" : "OBS",
    "task_name" : "newtask",
    "create_time" : 1606554932552,
    "state" : "RUNNING",
    "last_transfer_timestamp" : 1606984428612
  } ],
  "total_number" : 1
}
```

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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class ListTransferTasksSolution {
```

```
    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        ListTransferTasksRequest request = new ListTransferTasksRequest();
        request.withStreamName("{stream_name}");
        try {
            ListTransferTasksResponse response = client.listTransferTasks(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 huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListTransferTasksRequest()
        request.stream_name = "{stream_name}"
```

```
        response = client.list_transfer_tasks(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"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListTransferTasksRequest{}
    request.StreamName = "{stream_name}"
    response, err := client.ListTransferTasks(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	正常返回

错误码

请参见**错误码**。

5.5.3 删除转储任务

功能介绍

该接口用于删除转储任务。

调用方法

请参见**如何调用API**。

URI

DELETE /v2/{project_id}/streams/{stream_name}/transfer-tasks/{task_name}

表 5-89 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
stream_name	是	String	已创建的通道的名称。
task_name	是	String	待删除的转储任务名称。

请求参数

表 5-90 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

无

请求示例

删除转储任务

```
DELETE https://{Endpoint}/v2/{project_id}/streams/{stream_name}/transfer-tasks/{task_name}
```

响应示例

无

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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class DeleteTransferTaskSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        DeleteTransferTaskRequest request = new DeleteTransferTaskRequest();
        request.withStreamName("{stream_name}");
        request.withTaskName("{task_name}");
        try {
            DeleteTransferTaskResponse response = client.deleteTransferTask(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 huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteTransferTaskRequest()
        request.stream_name = "{stream_name}"
        request.task_name = "{task_name}"
        response = client.delete_transfer_task(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"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())
```

```
    request := &model.DeleteTransferTaskRequest{}
    request.StreamName = "{stream_name}"
    request.TaskName = "{task_name}"
    response, err := client.DeleteTransferTask(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
204	正常返回

错误码

请参见**错误码**。

5.5.4 查询转储任务详情

功能介绍

查询转储任务详情。

调用方法

请参见**如何调用API**。

URI

GET /v2/{project_id}/streams/{stream_name}/transfer-tasks/{task_name}

表 5-91 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
stream_name	是	String	已创建的通道的名称。
task_name	是	String	待删除的转储任务名称。

请求参数

表 5-92 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 5-93 响应 Body 参数

参数	参数类型	描述
stream_name	String	该转储任务所属通道名称。
task_name	String	转储任务名称。
state	String	转储任务状态。 ● ERROR：错误。 ● STARTING：启动中。 ● PAUSED：已停止。 ● RUNNING：运行中。 ● DELETE：已删除。 ● ABNORMAL：异常。
destination_type	String	转储任务类型。 ● OBS：转储到OBS。 ● MRS：转储到MRS。 ● DLI：转储到DLI。 ● CLOUDTABLE：转储到CloudTable。 ● DWS：转储到DWS。
create_time	Long	转储任务创建时间。
last_transfer_timestamp	Long	转储任务最近一次转储时间。
partitions	Array of **PartitionResult** objects	分区转储详情列表。

参数	参数类型	描述
obs_destination_description	**OBSDestinationDescriptorRequest** object	转储目的地为OBS的参数列表。
dws_destination_descripton	**DWSDestinationDescriptorRequest** object	转储目的地为DWS的参数列表。
mrs_destination_description	**MRSDestinationDescriptorRequest** object	转储目的地为MRS的参数列表。
dli_destination_description	**DliDestinationDescriptorRequest** object	转储目的地为DLI的参数列表。
cloudtable_destination_descripton	**CloudtableDestinationDescriptorRequest** object	转储目的地为CloudTable的参数列表。

表 5-94 PartitionResult

参数	参数类型	描述
status	String	分区的当前状态。 • CREATING：创建中 • ACTIVE：可用 • DELETED：删除中 • EXPIRED：已过期
partition_id	String	分区的唯一标识符。
hash_range	String	分区的可能哈希键值范围。
sequence_number_range	String	分区的序列号范围。
parent_partitions	String	父分区。

表 5-95 OBSDestinationDescriptorRequest

参数	参数类型	描述
task_name	String	转储任务的名称。 任务名称由英文字母、数字、中划线和下划线组成。长度为1～64个字符。

参数	参数类型	描述
agency_name	String	在统一身份认证服务(IAM)中创建委托的名称，DIS需要获取IAM委托信息去访问您指定的资源。创建委托的参数设置如下：- 委托类型：云服务- 云服务：DIS-持续时间：永久-“所属区域”为“全局服务”，“项目”为“对象存储服务”对应的“策略”包含“Tenant Administrator”。如果已经创建过委托，可以使用IAM服务提供的查询委托列表接口，获取有效可用的委托名称。取值范围：长度不超过64位，且不可配置为空。如果有在Console控制台使用转储任务，会提示自动创建委托，自动创建的委托名称为：dis_admin_agency
deliver_time_interval	Integer	根据用户配置的时间，周期性的将数据导入OBS，若某个时间段内无数据，则此时间段不会生成打包文件。 单位：秒
consumer_strategy	String	偏移量。 ● LATEST：最大偏移量，即获取最新的数据。 ● TRIM_HORIZON：最小偏移量，即读取最早的数据。
file_prefix	String	在OBS中存储通道文件的自定义目录，多级目录可用“/”进行分隔，不可以“/”开头。 取值范围：英文字母、数字、下划线和斜杠，最大长度为50个字符。 默认配置为空。
partition_format	String	将转储文件的生成时间使用“yyyy/MM/dd/HH/mm”格式生成分区字符串，用来定义写到OBS的Object文件所在的目录层次结构。- N/A：置空，不使用日期时间目录。- yyyy：年-yyyy/MM：年/- yyyy/MM/dd：年/月/日- yyyy/MM/dd/HH：年/月/日/时-yyyy/MM/dd/HH/mm：年/月/日/时/分例如：2017/11/10/14/49，目录结构就是“2017 > 11 > 10 > 14 > 49”，“2017”表示最外层文件夹。默认值：空说明：数据转储成功后，存储的目录结构为“obs_bucket_path/file_prefix/partition_format”。
obs_bucket_path	String	存储该通道数据的OBS桶名称。

参数	参数类型	描述
destination_file_type	String	转储文件格式。 ● text：转储目标格式为TEXT，缺省值 ● parquet：转储目标格式为Parquet ● carbon：转储目标格式为Carbon 说明： “源数据类型”为“JSON”，“转储服务类型”为“OBS”时才可选择“parquet”或“carbon”格式。
processing_schema	ProcessingSchema object	根据源数据的时间戳和已配置的"partition_format"生成对应的转储时间目录。将源数据的时间戳使用“yyyy/MM/dd/HH/mm”格式生成分区字符串，用来定义写到OBS的Object文件所在的目录层次结构。
record_delimiter	String	转储文件的记录分隔符，用于分隔写入转储文件的用户数据。 取值范围： ● 逗号 ","，默认值 ● 分号 ";" ● 竖线 "|" ● 换行符 "\n"

表 5-96 ProcessingSchema

参数	参数类型	描述
timestamp_name	String	源数据时间戳的属性名称。
timestamp_type	String	源数据时间戳的类型。 ● String ● Timestamp：Long类型的13位时间戳
timestamp_format	String	源数据时间戳的类型为String时必选，用于根据时间戳格式生成OBS的时间目录。

表 5-97 DWSDestinationDescriptorRequest

参数	参数类型	描述
task_name	String	转储任务的名称。 任务名称由英文字母、数字、中划线和下划线组成。长度为1～64个字符。
agency_name	String	在统一身份认证服务(IAM)中创建委托的名称，DIS需要获取IAM委托信息去访问您指定的资源。创建委托的参数设置如下：- 委托类型：云服务- 云服务：DIS- 持续时间：永久-“所属区域”为“全局服务”，“项目”为“对象存储服务”对应的“策略”包含“Tenant Administrator”。如果已经创建过委托，可以使用IAM服务提供的查询委托列表接口，获取有效可用的委托名称。取值范围：长度不超过64位，且不可配置为空。如果有在Console控制台使用转储任务，会提示自动创建委托，自动创建的委托名称为：dis_admin_agency
deliver_time_interval	Integer	根据用户配置的时间，周期性的将数据导入OBS，若某个时间段内无数据，则此时间段不会生成打包文件。 单位：秒
consumer_strategy	String	偏移量。 ● LATEST：最大偏移量，即获取最新的数据。 ● TRIM_HORIZON：最小偏移量，即读取最早的数据。
dws_cluster_name	String	存储该通道数据的DWS集群名称。
dws_cluster_id	String	存储该通道数据的DWS集群ID。
dws_database_name	String	存储该通道数据的DWS数据库名称。
dws_schema	String	存储该通道数据的DWS数据库模式。
dws_table_name	String	存储该通道数据的DWS数据库模式下的数据表。
dws_delimiter	String	用户数据的字段分隔符，根据此分隔符分隔用户数据插入DWS数据表的相应列。 取值范围：“，”、“；”和“|”三种字符中的一个。
user_name	String	存储该通道数据的DWS数据库的用户名。

参数	参数类型	描述
user_password	String	存储该通道数据的DWS数据库的密码。
kms_user_key_name	String	用户在密钥管理服务（简称KMS）创建的用户主密钥名称，用于加密存储DWS数据库的密码。
kms_user_key_id	String	用户在密钥管理服务（简称KMS）创建的用户主密钥ID，用于加密存储DWS数据库的密码。
obs_bucket_path	String	临时存储该通道数据的OBS桶名称。
file_prefix	String	临时存储该通道数据的OBS桶下的自定义目录，多级目录可用“/”进行分隔，不可以“/”开头。 取值范围：英文字母、数字、下划线和斜杠，最大长度为50个字符。 默认配置为空。
retry_duration	String	用户数据导入DWS集群失败的重试失效时间。超出此配置项配置的时间，转储DWS失败的数据将备份至“OBS桶/file_prefix/dws_error”目录下。取值范围： 0～7200。单位：秒。默认配置为1800。
dws_table_columns	String	指定要转储到DWS表中的列，为null或者为空则默认全列。比如“c1,c2”表示将Schema中c1和c2这两列转储到DWS。 默认为空。
options	**Options** object	DWS容错性选项（用于指定外表数据的各类参数）。

表 5-98 Options

参数	参数类型	描述
fill_missing_fields	String	数据入库时，数据源文件中某行的最后一个字段缺失时，请选择是直接将字段设为Null，还是在错误表中报错提示。 取值范围： ● true/on ● false/off 缺省值：false/off

参数	参数类型	描述
ignore_extra_data	String	数据源文件中的字段比外表定义列数多时，是否忽略多出的列。该参数只在数据导入过程中使用。 取值范围： ● true/on ● false/off 缺省值：false/off
compatible_illegal_chars	String	导入非法字符容错参数。是将非法字符按照转换规则转换后入库，还是报错中止导入。 取值范围： ● true/on ● false/off 缺省值：false/off
reject_limit	String	指定本次数据导入允许出现的数据格式错误个数，当导入过程中出现的数据格式错误未达到限定值时，本次数据导入可以成功。 取值范围： ● 整型值 ● unlimited（无限制） 缺省值为0，有错误信息立即返回。
error_table_name	String	用于记录数据格式错误信息的错误表表名。并行导入结束后查询此错误信息表，能够获取详细的错误信息。

表 5-99 MRSDestinationDescriptorRequest

参数	参数类型	描述
task_name	String	转储任务的名称。 任务名称由英文字母、数字、中划线和下划线组成。长度为1～64个字符。

参数	参数类型	描述
agency_name	String	在统一身份认证服务(IAM)中创建委托的名称，DIS需要获取IAM委托信息去访问您指定的资源。创建委托的参数设置如下：- 委托类型：云服务- 云服务：DIS-持续时间：永久-“所属区域”为“全局服务”，“项目”为“对象存储服务”对应的“策略”包含“Tenant Administrator”。如果已经创建过委托，可以使用IAM服务提供的查询委托列表接口，获取有效可用的委托名称。取值范围：长度不超过64位，且不可配置为空。如果有在Console控制台使用转储任务，会提示自动创建委托，自动创建的委托名称为：dis_admin_agency
deliver_time_interval	Integer	根据用户配置的时间，周期性的将数据导入OBS，若某个时间段内无数据，则此时间段不会生成打包文件。 单位：秒
consumer_strategy	String	偏移量。 ● LATEST：最大偏移量，即获取最新的数据。 ● TRIM_HORIZON：最小偏移量，即读取最早的数据。
mrs_cluster_name	String	存储该通道数据的MRS集群名称。 说明： 仅支持非Kerberos认证的MRS集群。
mrs_cluster_id	String	存储该通道数据的MRS集群ID。
mrs_hdfs_path	String	存储该通道数据的MRS集群的HDFS路径。
file_prefix	String	临时存储该通道数据的OBS桶下的自定义目录，多级目录可用“/”进行分隔，不可以“/”开头。 取值范围：英文字母、数字、下划线和斜杠，最大长度为50个字符。 默认配置为空。
hdfs_prefix_folder	String	在MRS集群HDFS中存储通道文件的自定义目录，多级目录可用"/"进行分隔。 取值范围：0~50个字符。 默认配置为空。
obs_bucket_path	String	临时存储该通道数据的OBS桶名称。

参数	参数类型	描述
retry_duration	String	用户数据转储失败的失效重试时间。重试时间超过该配置项配置的值，则将转储失败的数据备份至“OBS桶/file_prefix/mrs_error”目录下。 取值范围：0~7200。 单位：秒。 默认配置为1800。 配置为“0”表示DIS服务不会在转储失败时进行重试。

表 5-100 DliDestinationDescriptorRequest

参数	参数类型	描述
task_name	String	转储任务的名称。 任务名称由英文字母、数字、中划线和下划线组成。长度为1～64个字符。
agency_name	String	在统一身份认证服务(IAM)中创建委托的名称，DIS需要获取IAM委托信息去访问您指定的资源。创建委托的参数设置如下：- 委托类型：云服务- 云服务：DIS- 持续时间：永久- “所属区域”为“全局服务”，“项目”为“对象存储服务”对应的“策略”包含“Tenant Administrator”。如果已经创建过委托，可以使用IAM服务提供的查询委托列表接口，获取有效可用的委托名称。取值范围：长度不超过64位，且不可配置为空。如果有在Console控制台使用转储任务，会提示自动创建委托，自动创建的委托名称为：dis_admin_agency
deliver_time_interval	Integer	根据用户配置的时间，周期性的将数据导入OBS，若某个时间段内无数据，则此时间段不会生成打包文件。 单位：秒
consumer_strategy	String	偏移量。 ● LATEST：最大偏移量，即获取最新的数据。 ● TRIM_HORIZON：最小偏移量，即读取最早的数据。
dli_database_name	String	存储该通道数据的DLI数据库名称。

参数	参数类型	描述
dli_table_name	String	存储该通道数据的DLI表名称。 说明： 仅支持数据位置为DLI的表，且用户需具有该表的插入权限。
obs_bucket_path	String	临时存储该通道数据的OBS桶名称。
file_prefix	String	临时存储该通道数据的OBS桶下的自定义目录，多级目录可用“/”进行分隔，不可以“/”开头。 取值范围：英文字母、数字、下划线和斜杠，最大长度为50个字符。 默认配置为空。
retry_duration	String	用户数据导入DLI失败的失效重试时间。重试时间超过该配置项配置的值，则将转储失败的数据备份至“OBS桶/file_prefix/dli_error”目录下。取值范围：0~7200。单位：秒。默认配置为1800。配置为“0”表示DIS服务不会在转储失败时进行重试。

表 5-101 CloudtableDestinationDescriptorRequest

参数	参数类型	描述
task_name	String	转储任务的名称。 任务名称由英文字母、数字、中划线和下划线组成。长度为1～64个字符。
agency_name	String	在统一身份认证服务(IAM)中创建委托的名称，DIS需要获取IAM委托信息去访问您指定的资源。创建委托的参数设置如下：- 委托类型：云服务- 云服务：DIS- 持续时间：永久- “所属区域”为“全局服务”，“项目”为“对象存储服务”对应的“策略”包含“Tenant Administrator”。如果已经创建过委托，可以使用IAM服务提供的查询委托列表接口，获取有效可用的委托名称。取值范围：长度不超过64位，且不可配置为空。如果有在Console控制台使用转储任务，会提示自动创建委托，自动创建的委托名称为：dis_admin_agency
deliver_time_interval	Integer	根据用户配置的时间，周期性的将数据导入OBS，若某个时间段内无数据，则此时间段不会生成打包文件。 单位：秒

参数	参数类型	描述
consumer_strategy	String	偏移量。 • LATEST：最大偏移量，即获取最新的数据。 • TRIM_HORIZON：最小偏移量，即读取最早的数据。
cloudtable_cluster_name	String	存储该通道数据的CloudTable集群名称。 如果选择转储OpenTSDB，则集群必须开启OpenTSDB。
cloudtable_cluster_id	String	存储该通道数据的CloudTable集群ID。 如果选择转储OpenTSDB，则集群必须开启OpenTSDB。
cloudtable_table_name	String	转储HBase时必选，表示存储该通道数据的CloudTable集群HBase表名称。
cloudtable_schema	**CloudtableSchema** object	转储HBase时必选，与“opentsdb_schema”二选一，表示CloudTable集群HBase数据的Schema配置。用于将通道内的JSON数据进行格式转换并导入Cloudtable的HBase表中。
opentsdb_schema	Array of **OpenTSDBSchema** objects	转储OpenTSDB时必选，与“cloudtable_schema”二选一，表示CloudTable集群OpenTSDB数据的Schema配置。用于将通道内的JSON数据进行格式转换并导入Cloudtable的OpenTSDB。
cloudtable_row_key_delimiter	String	转储HBase的rowkey分隔符，用于分隔生成rowKey的用户数据。取值范围：”,”、”.”、”|”、”;”、”\”、”-”、”_”、”~”缺省值：”.”
obs_backup_bucket_path	String	用户数据转储CloudTable服务失败时，可选择将转储失败的数据备份至OBS服务，此参数为OBS服务的桶名称。
backup_file_prefix	String	用户数据转储CloudTable服务失败时，可选择将转储失败的数据备份至OBS服务，此参数为OBS桶下的自定义目录，多级目录可用“/”进行分隔，不可以“/”开头。取值范围：英文字母、数字和下划线。最大长度：最大长度为50个字符。默认配置为空。

参数	参数类型	描述
retry_duration	String	用户数据导入CloudTable服务失败的失效重试时间。超出此时效，转储CloudTable失败的数据将备份至“OBS桶/ backup_file_prefix / cloudtable_error”或“OBS桶/ backup_file_prefix /opentsdb_error”目录下。取值范围： 0～7200。单位：秒。默认配置为1800。

表 5-102 CloudtableSchema

参数	参数类型	描述
row_key	Array of **RowKey** objects	CloudTable集群HBase数据rowkey的Schema配置，用于将通道内的JSON数据生成HBase数据的rowkey。 取值范围：1～64。
columns	Array of **Column** objects	CloudTable集群HBase数据列的Schema配置，用于将通道内的JSON数据生成HBase数据的列。 取值范围：1～4096。

表 5-103 RowKey

参数	参数类型	描述
value	String	通道内JSON数据的JSON属性名，用于生成HBase数据的rowkey。
type	String	通道内JSON数据的JSON属性的类型名称。

表 5-104 Column

参数	参数类型	描述
column_family_name	String	存储该通道数据的HBase表数据的列族名称。
column_name	String	存储该通道数据的HBase表数据的列名称。 取值范围：1～32，只能包含英文字母、数字和下划线。

参数	参数类型	描述
value	String	通道内JSON数据的JSON属性名，用于生成HBase数据的列值。
type	String	通道内JSON数据的JSON属性的类型名称。

表 5-105 OpenTSDBSchema

参数	参数类型	描述
metric	Array of **OpenTSDBMetric** objects	CloudTable集群OpenTSDB数据metric的Schema配置，用于将通道内的JSON数据进行格式转换生成OpenTSDB数据的metric。
timestamp	**OpenTSDBTimestamp** object	CloudTable集群OpenTSDB 数据timestamp的Schema配置，用于将通道内的JSON数据进行格式转换生成OpenTSDB数据的timestamp。
value	**OpenTSDBValue** object	CloudTable集群OpenTSDB 数据value的Schema配置，用于将通道内的JSON数据进行格式转换生成OpenTSDB 数据的value。
tags	Array of **OpenTSDBTags** objects	CloudTable集群OpenTSDB数据tags的Schema配置，用于将通道内的JSON数据进行格式转换生成OpenTSDB数据的tags。

表 5-106 OpenTSDBMetric

参数	参数类型	描述
type	String	• Constant表示metric为常量value的值。 • String表示metric为通道内用户数据对应JSON属性的取值，且该JOSN属性的取值为String。
value	String	常量或通道内用户数据的JSON属性名称。 取值范围：1～32，只能包含英文字母、数字和点。

表 5-107 OpenTSDBTimestamp

参数	参数类型	描述
type	String	● Timestamp类型表示通道内用户数据对应JSON属性的取值为Timestamp类型，不需要进行数据格式转换就可以生成OpenTSDB的timestamp。- String类型表示通道内用户数据对应JSON属性的取值为Date格式，需要进行数据格式转换才能生成OpenTSDB的timestamp。
value	String	通道内用户数据的JSON属性名称。 取值范围：1～32，只能包含英文字母、数字和下划线。
format	String	“type”为“String”类型时必选。表示通道内用户数据对应JSON属性的取值为Date格式，需要根据format字段进行数据格式转换生成OpenTSDB的timestamp。取值范围：- yyyy/MM/dd HH:mm:ss- MM/dd/yyyy HH:mm:ss- dd/MM/yyyy HH:mm:ss- yyyy-MM-dd HH:mm:ss- MM-dd-yyyy HH:mm:ss- dd-MM-yyyy HH:mm:ss

表 5-108 OpenTSDBValue

参数	参数类型	描述
type	String	通道内用户JSON数据对应JSON属性的类型名称。 取值范围： ● Bigint ● Double ● Boolean ● Timestamp ● String ● Decimal
value	String	常量或通道内用户数据的JSON属性名称。 取值范围：1～32，只能包含英文字母、数字和下划线。

表 5-109 OpenTSDBTags

参数	参数类型	描述
name	String	存储该通道数据的OpenTSDB数据的tag名称。 取值范围：1～32，只能包含英文字母、数字和下划线。
type	String	通道内用户JSON数据对应JSON属性的类型名称。 取值范围： ● Bigint ● Double ● Boolean ● Timestamp ● String ● Decimal
value	String	常量或通道内用户数据的JSON属性名称。 取值范围：1～32，只能包含英文字母、数字和下划线。

请求示例

查询转储任务详情

```
GET https://{Endpoint}/v2/{project_id}/streams/{stream_name}/transfer-tasks/{task_name}
```

响应示例

状态码：200

正常返回

```
{
  "stream_id" : "RdMFID6edQdf8eDzc9e",
  "stream_name" : "newstream",
  "task_name" : "newtask",
  "task_id" : "As805BudhcH1lDs6gbn",
  "destination_type" : "OBS",
  "state" : "RUNNING",
  "create_time" : 1606554932552,
  "last_transfer_timestamp" : 1606984428612,
  "obs_destination_description" : {
    "agency_name" : "dis_admin_agency",
    "file_prefix\"" : "",
    "partition_format" : "yyyy/MM/dd",
    "obs_bucket_path" : "obsbucket",
    "deliver_time_interval" : 60,
    "consumer_strategy" : "LATEST",
    "retry_duration" : 0,
    "destination_file_type" : "text",
    "record_delimiter" : "\n\n"
```

```
 },
 "partitions" : [ {
  "partitionId" : "shardId-0000000000",
  "discard" : 0,
  "state" : "RUNNING",
  "last_transfer_timestamp" : 1606984428612,
  "last_transfer_offset" : 289897
 } ]
}
```

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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class ShowTransferTaskSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowTransferTaskRequest request = new ShowTransferTaskRequest();
        request.withStreamName("{stream_name}");
        request.withTaskName("{task_name}");
        try {
            ShowTransferTaskResponse response = client.showTransferTask(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 huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowTransferTaskRequest()
        request.stream_name = "{stream_name}"
        request.task_name = "{task_name}"
        response = client.show_transfer_task(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"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
```

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

    request := &model.ShowTransferTaskRequest{}
    request.StreamName = "{stream_name}"
    request.TaskName = "{task_name}"
    response, err := client.ShowTransferTask(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	正常返回

错误码

请参见**错误码**。

5.5.5 批量启动转储任务

功能介绍

此接口用于批量启动转储任务。

调用方法

请参见**如何调用API**。

URI

POST /v2/{project_id}/streams/{stream_name}/transfer-tasks/action

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。
stream_name	是	String	需要查询的通道名称。

请求参数

表 5-111 请求 Body 参数

参数	是否必选	参数类型	描述
action	是	String	转储任务操作，目前支持： ● start：启动转储任务
tasks	是	Array of **BatchTransferTask** objects	待操作的转储任务列表。

表 5-112 BatchTransferTask

参数	是否必选	参数类型	描述
id	是	String	转储任务ID

响应参数

状态码：200

表 5-113 响应 Body 参数

参数	参数类型	描述
-	String	-

请求示例

批量启动转储任务

```
POST https://{Endpoint}/v2/{project_id}/streams/{stream_name}/transfer-tasks/action

{
  "action" : "start",
  "tasks" : [ {
    "id" : "9dSu1wfCytSk1aOLxvF"
  } ]
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

批量启动转储任务

```
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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;

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

public class BatchStartTransferTaskSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        BatchStartTransferTaskRequest request = new BatchStartTransferTaskRequest();
        request.withStreamName("{stream_name}");
        BatchStartTransferTaskReq body = new BatchStartTransferTaskReq();
        List<BatchTransferTask> listbodyTasks = new ArrayList<>();
        listbodyTasks.add(
            new BatchTransferTask()
                .withId("9dSu1wfCytSk1aOLxvF")
        );
        body.withTasks(listbodyTasks);
        body.withAction(BatchStartTransferTaskReq.ActionEnum.fromValue("start"));
        request.withBody(body);
        try {
            BatchStartTransferTaskResponse response = client.batchStartTransferTask(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

批量启动转储任务

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = BatchStartTransferTaskRequest()
        request.stream_name = "{stream_name}"
        listTasksbody = [
            BatchTransferTask(
                id="9dSu1wfCytSk1aOLxvF"
            )
        ]
        request.body = BatchStartTransferTaskReq(
            tasks=listTasksbody,
            action="start"
        )
        response = client.batch_start_transfer_task(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

批量启动转储任务

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.BatchStartTransferTaskRequest{}
    request.StreamName = "{stream_name}"
    var listTasksbody = []model.BatchTransferTask{
        {
            Id: "9dSu1wfCytSk1aOLxvF",
        },
    }
    request.Body = &model.BatchStartTransferTaskReq{
        Tasks: listTasksbody,
        Action: model.GetBatchStartTransferTaskReqActionEnum().START,
    }
    response, err := client.BatchStartTransferTask(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	正常返回

错误码

请参见**错误码**。

5.5.6 批量暂停转储任务

功能介绍

此接口用于批量暂停转储任务。

调用方法

请参见**如何调用API**。

URI

POST /v2/{project_id}/streams/{stream_name}/transfer-tasks/action

表 5-114 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
stream_name	是	String	需要查询的通道名称。

请求参数

表 5-115 请求 Body 参数

参数	是否必选	参数类型	描述
action	是	String	转储任务操作，目前支持： • stop：停止转储任务
tasks	是	Array of **BatchTransferTask** objects	待暂停的转储任务列表。

表 5-116 BatchTransferTask

参数	是否必选	参数类型	描述
id	是	String	转储任务ID

响应参数

无

请求示例

批量暂停转储任务

```
POST https://{Endpoint}/v2/{project_id}/streams/{stream_name}/transfer-tasks/action

{
  "action" : "stop",
  "tasks" : [ {
    "id" : "9dSu1wfCytSk1aOLxvF"
  } ]
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

批量暂停转储任务

```
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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;

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

public class BatchStopTransferTaskSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        BatchStopTransferTaskRequest request = new BatchStopTransferTaskRequest();
        request.withStreamName("{stream_name}");
        BatchStopTransferTaskReq body = new BatchStopTransferTaskReq();
        List<BatchTransferTask> listbodyTasks = new ArrayList<>();
        listbodyTasks.add(
            new BatchTransferTask()
                .withId("9dSu1wfCytSk1aOLxvF")
        );
        body.withTasks(listbodyTasks);
        body.withAction(BatchStopTransferTaskReq.ActionEnum.fromValue("stop"));
        request.withBody(body);
        try {
            BatchStopTransferTaskResponse response = client.batchStopTransferTask(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

批量暂停转储任务

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = BatchStopTransferTaskRequest()
        request.stream_name = "{stream_name}"
        listTasksbody = [
            BatchTransferTask(
                id="9dSu1wfCytSk1aOLxvF"
            )
        ]
        request.body = BatchStopTransferTaskReq(
            tasks=listTasksbody,
            action="stop"
        )
        response = client.batch_stop_transfer_task(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

批量暂停转储任务

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.BatchStopTransferTaskRequest{}
    request.StreamName = "{stream_name}"
    var listTasksbody = []model.BatchTransferTask{
        {
            Id: "9dSu1wfCytSk1aOLxvF",
        },
    }
    request.Body = &model.BatchStopTransferTaskReq{
        Tasks: listTasksbody,
        Action: model.GetBatchStopTransferTaskReqActionEnum().STOP,
    }
    response, err := client.BatchStopTransferTask(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	正常返回

错误码

请参见**错误码**。

5.6 监控管理

5.6.1 查询通道监控

功能介绍

本接口用于查询指定通道的监控数据。

调用方法

请参见**如何调用API**。

URI

GET /v2/{project_id}/streams/{stream_name}/metrics

表 5-117 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
stream_name	是	String	通道名称。

表 5-118 Query 参数

参数	是否必选	参数类型	描述
label	否	String	通道监控指标。（label与label_list必须二选一，label_list与label同时存在时，以label_list为准） • total_put_bytes_per_stream：总输入流量（Byte） • total_get_bytes_per_stream：总输出流量（Byte） • total_put_records_per_stream：总输入记录数（个） • total_get_records_per_stream：总输出记录数（个） • total_put_req_latency：上传请求平均处理时间（毫秒） • total_get_req_latency：下载请求平均处理时间（毫秒） • total_put_req_suc_per_stream：上传请求成功次数（个） • total_get_req_suc_per_stream：下载请求成功次数（个） • traffic_control_put：因流控拒绝的上传请求次数（个） • traffic_control_get：因流控拒绝的下载请求次数（个）
label_list	否	String	使用label用逗号拼接组成，用于批量查询多个label的指标。（label与label_list必须二选一，label_list与label同时存在时，以label_list为准）
start_time	是	Long	监控开始时间点，10位时间戳。
end_time	是	Long	监控结束时间点，10位时间戳。

请求参数

表 5-119 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 5-120 响应 Body 参数

参数	参数类型	描述
metrics	**Metrics** object	数据对象。
metrics_list	Array of **Metrics** objects	监控数据对象列表。

表 5-121 Metrics

参数	参数类型	描述
dataPoints	Array of **DataPoint** objects	监控数据。
label	String	监控指标。

表 5-122 DataPoint

参数	参数类型	描述
timestamp	Long	时间戳。
value	Long	时间戳对应的监控值。

请求示例

查询通道监控数据

```
GET https://{Endpoint}/v2/{project_id}/streams/{stream_name}/metrics
```

响应示例

无

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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class ShowStreamMetricsSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowStreamMetricsRequest request = new ShowStreamMetricsRequest();
        request.withStreamName("{stream_name}");
        try {
            ShowStreamMetricsResponse response = client.showStreamMetrics(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 huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowStreamMetricsRequest()
        request.stream_name = "{stream_name}"
        response = client.show_stream_metrics(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"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowStreamMetricsRequest{}
    request.StreamName = "{stream_name}"
```

```
    response, err := client.ShowStreamMetrics(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	正常返回

错误码

请参见**错误码**。

5.6.2 查询分区监控

功能介绍

本接口用于查询通道指定分区的监控数据。

调用方法

请参见**如何调用API**。

URI

GET /v2/{project_id}/streams/{stream_name}/partitions/{partition_id}/metrics

表 5-123 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
stream_name	是	String	通道名称。
partition_id	是	String	分区编号。可定义为如下两种样式：- shardId-0000000000- 0比如一个通道有三个分区，那么分区标识符分别为0, 1, 2，或者shardId-0000000000, shardId-0000000001, shardId-0000000002

表 5-124 Query 参数

参数	是否必选	参数类型	描述
label	否	String	分区监控指标。（label与label_list必须二选一，label_list与label同时存在时，以label_list为准） • total_put_bytes_per_partition：分区总输入流量（Byte） • total_get_bytes_per_partition：分区总输出流量（Byte） • total_put_records_per_partition：分区总输入记录数（个） • total_get_records_per_partition：分区总输出记录数（个）
label_list	否	String	使用label用逗号拼接组成，用于批量查询多个label的指标。（label与label_list必须二选一，label_list与label同时存在时，以label_list为准）
start_time	是	Long	监控开始时间点，10位时间戳。
end_time	是	Long	监控结束时间点，10位时间戳。

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）。

响应参数

状态码：200

表 5-126 响应 Body 参数

参数	参数类型	描述
metrics	**Metrics** object	数据对象。

表 5-127 Metrics

参数	参数类型	描述
dataPoints	Array of **DataPoint** objects	监控数据。
label	String	监控指标。

表 5-128 DataPoint

参数	参数类型	描述
timestamp	Long	时间戳。
value	Long	时间戳对应的监控值。

请求示例

查询分区监控数据

```
GET https://{Endpoint}/v2/{project_id}/streams/{stream_name}/partitions/{partition_id}/metrics
```

响应示例

无

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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class ShowPartitionMetricsSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowPartitionMetricsRequest request = new ShowPartitionMetricsRequest();
        request.withStreamName("{stream_name}");
        request.withPartitionId("{partition_id}");
        try {
            ShowPartitionMetricsResponse response = client.showPartitionMetrics(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 huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowPartitionMetricsRequest()
        request.stream_name = "{stream_name}"
        request.partition_id = "{partition_id}"
```

```
        response = client.show_partition_metrics(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"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowPartitionMetricsRequest{}
    request.StreamName = "{stream_name}"
    request.PartitionId = "{partition_id}"
    response, err := client.ShowPartitionMetrics(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	正常返回

错误码

请参见**错误码**。

5.7 标签管理

5.7.1 给指定通道添加标签

功能介绍

本接口用于给指定通道添加标签。

调用方法

请参见**如何调用API**。

URI

POST /v2/{project_id}/stream/{stream_id}/tags

表 5-129 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
stream_id	是	String	通道ID。

请求参数

表 5-130 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 5-131 请求 Body 参数

参数	是否必选	参数类型	描述
tag	是	**Tag** object	标签对象。

表 5-132 Tag

参数	是否必选	参数类型	描述
key	否	String	键。 ● 不能为空。 ● 对于同一资源键值唯一。 ● 字符集：A-Z，a-z ， 0-9，‘-’，‘_’，UNICODE字符（\u4E00-\u9FFF）。
value	否	String	值。 ● 长度不超过43个字符。 ● 字符集：A-Z，a-z ， 0-9，‘.’，‘-’，‘_’，UNICODE字符（\u4E00-\u9FFF）。 ● 只能包含数字、字母、中划线“-”、下划线“_”。

响应参数

无

请求示例

给指定通道添加标签

```
POST https://{Endpoint}/v2/{project_id}/stream/{stream_id}/tags

{
  "tag" : {
    "key" : "key",
    "value" : "value"
  }
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

给指定通道添加标签

```
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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;


public class CreateTagSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateTagRequest request = new CreateTagRequest();
        request.withStreamId("{stream_id}");
        CreateTagReq body = new CreateTagReq();
        Tag tagbody = new Tag();
        tagbody.withKey("key")
            .withValue("value");
        body.withTag(tagbody);
        request.withBody(body);
        try {
            CreateTagResponse response = client.createTag(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

给指定通道添加标签

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateTagRequest()
        request.stream_id = "{stream_id}"
        tagbody = Tag(
            key="key",
            value="value"
        )
        request.body = CreateTagReq(
            tag=tagbody
        )
        response = client.create_tag(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

给指定通道添加标签

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateTagRequest{}
```

```
    request.StreamId = "{stream_id}"
    keyTag:= "key"
    valueTag:= "value"
    tagbody := &model.Tag{
        Key: &keyTag,
        Value: &valueTag,
    }
    request.Body = &model.CreateTagReq{
        Tag: tagbody,
    }
    response, err := client.CreateTag(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
204	正常返回

错误码

请参见**错误码**。

5.7.2 查询指定通道的标签信息

功能介绍

该接口用于查询指定通道的标签信息。

调用方法

请参见**如何调用API**。

URI

GET /v2/{project_id}/stream/{stream_id}/tags

表 5-133 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
stream_id	是	String	通道ID。

请求参数

表 5-134 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 5-135 响应 Body 参数

参数	参数类型	描述
tags	Array of **Tag** objects	标签列表。

表 5-136 Tag

参数	参数类型	描述
key	String	键。 ● 不能为空。 ● 对于同一资源键值唯一。 ● 字符集：A-Z，a-z， 0-9，‘-’，‘_’，UNICODE字符（\u4E00-\u9FFF）。
value	String	值。 ● 长度不超过43个字符。 ● 字符集：A-Z，a-z， 0-9，‘.’，‘-’，‘_’，UNICODE字符（\u4E00-\u9FFF）。 ● 只能包含数字、字母、中划线“-”、下划线“_”。

请求示例

查询指定通道的标签信息。

```
GET https://{Endpoint}/v2/{project_id}/stream/{stream_id}/tags
```

响应示例

状态码：200

通道标签信息响应体。

```
{
  "tags" : [ {
    "key" : "key1",
    "value" : "value1"
  }, {
    "key" : "key2",
    "value" : "value3"
  } ]
}
```

状态码

状态码	描述
200	通道标签信息响应体。

错误码

请参见**错误码**。

5.7.3 删除指定通道的标签

功能介绍

该接口用于删除指定通道的标签。

调用方法

请参见**如何调用API**。

URI

DELETE /v2/{project_id}/stream/{stream_id}/tags/{key}

表 5-137 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
stream_id	是	String	通道ID。
key	是	String	标签键。

请求参数

表 5-138 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

无

请求示例

删除指定通道的标签。

```
DELETE https://{Endpoint}/v2/{project_id}/stream/{stream_id}/tags/{key}
```

响应示例

无

状态码

状态码	描述
204	正常返回

错误码

请参见**错误码**。

5.7.4 批量添加资源标签

功能介绍

该接口用于批量添加资源（通道等）标签。此接口为幂等接口：创建时如果请求体中存在重复key则报错。创建时，不允许设置重复key数据，如果数据库已存在该key，就覆盖value的值。

调用方法

请参见**如何调用API**。

URI

POST /v2/{project_id}/stream/{stream_id}/tags/action

表 5-139 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
stream_id	是	String	通道ID。

请求参数

表 5-140 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 5-141 请求 Body 参数

参数	是否必选	参数类型	描述
action	是	String	操作标识：仅限于create • create：批量创建
tags	是	Array of **Tag** objects	标签列表。

表 5-142 Tag

参数	是否必选	参数类型	描述
key	否	String	键。 • 不能为空。 • 对于同一资源键值唯一。 • 字符集：A-Z，a-z ， 0-9，‘-’，‘_’，UNICODE字符（\u4E00-\u9FFF）。

参数	是否必选	参数类型	描述
value	否	String	值。 ● 长度不超过43个字符。 ● 字符集：A-Z，a-z ， 0-9，‘.’，‘-’，‘_’，UNICODE字符（\u4E00-\u9FFF）。 ● 只能包含数字、字母、中划线“-”、下划线“_”。

响应参数

无

请求示例

批量添加资源标签

```
POST https://{Endpoint}/v2/{project_id}/stream/{stream_id}/tags/action

{
  "action" : "create",
  "tags" : [ {
    "key" : "key1",
    "value" : "value1"
  }, {
    "key" : "key2",
    "value" : "value3"
  } ]
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

批量添加资源标签

```
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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;

import java.util.List;
```

```
import java.util.ArrayList;

public class BatchCreateTagsSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        BatchCreateTagsRequest request = new BatchCreateTagsRequest();
        request.withStreamId("{stream_id}");
        BatchCreateTagsReq body = new BatchCreateTagsReq();
        List<Tag> listbodyTags = new ArrayList<>();
        listbodyTags.add(
            new Tag()
                .withKey("key1")
                .withValue("value1")
        );
        listbodyTags.add(
            new Tag()
                .withKey("key2")
                .withValue("value3")
        );
        body.withTags(listbodyTags);
        body.withAction(BatchCreateTagsReq.ActionEnum.fromValue("create"));
        request.withBody(body);
        try {
            BatchCreateTagsResponse response = client.batchCreateTags(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

批量添加资源标签

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = BatchCreateTagsRequest()
        request.stream_id = "{stream_id}"
        listTagsbody = [
            Tag(
                key="key1",
                value="value1"
            ),
            Tag(
                key="key2",
                value="value3"
            )
        ]
        request.body = BatchCreateTagsReq(
            tags=listTagsbody,
            action="create"
        )
        response = client.batch_create_tags(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

批量添加资源标签

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.BatchCreateTagsRequest{}
    request.StreamId = "{stream_id}"
    keyTags:= "key1"
    valueTags:= "value1"
    keyTags1:= "key2"
    valueTags1:= "value3"
    var listTagsbody = []model.Tag{
        {
            Key: &keyTags,
            Value: &valueTags,
        },
        {
            Key: &keyTags1,
            Value: &valueTags1,
        },
    }
    request.Body = &model.BatchCreateTagsReq{
        Tags: listTagsbody,
        Action: model.GetBatchCreateTagsReqActionEnum().CREATE,
    }
    response, err := client.BatchCreateTags(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
204	正常返回

错误码

请参见**错误码**。

5.7.5 查询指定区域所有标签集合

功能介绍

该接口用于查询指定区域所有标签集合。

调用方法

请参见**如何调用API**。

URI

GET /v2/{project_id}/stream/tags

表 5-143 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

请求参数

表 5-144 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 5-145 响应 Body 参数

参数	参数类型	描述
tags	Array of **Tags** objects	标签列表。

表 5-146 Tags

参数	参数类型	描述
key	String	键。 ● 不能为空。 ● 对于同一资源键值唯一。 ● 字符集：A-Z，a-z， 0-9， ‘-’，‘_’，UNICODE字符（\u4E00-\u9FFF）。

参数	参数类型	描述
values	Array of strings	标签值列表。 如果values为空列表，则表示any_value。value之间为或的关系。

请求示例

查询指定区域所有标签集合。

```
GET https://{Endpoint}/v2/{project_id}/stream/tags
```

响应示例

状态码：200

标签集合响应体。

```
{
  "tags" : [ {
    "key" : "key1",
    "values" : [ "value1", "value2" ]
  }, {
    "key" : "key2",
    "values" : [ "value1", "value2" ]
  } ]
}
```

状态码

状态码	描述
200	标签集合响应体。

错误码

请参见**错误码**。

5.7.6 使用标签过滤资源（通道等）

功能介绍

该接口用于使用标签过滤资源（通道等）。

调用方法

请参见**如何调用API**。

URI

POST /v2/{project_id}/stream/resource_instances/action

表 5-147 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

请求参数

表 5-148 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 5-149 请求 Body 参数

参数	是否必选	参数类型	描述
action	是	String	操作标识(仅限于filter，count) • filter：分页查询 • count：查询总条数，只需按照条件将总条数返回即可
limit	否	String	查询记录数(action为count时无此参数)如果action为filter默认为1000，limit最多为1000，不能为负数，最小值为1
offset	否	String	索引位置, 从offset指定的下一条数据开始查询。 查询第一页数据时，不需要传入此参数，查询后续页码数据时，将查询前一页数据时响应体中的值带入此参数(action为count时无此参数)如果action为filter默认为0，必须为数字，不能为负数
tags	否	Array of **Tags** objects	返回结果包含该参数中所有标签对应的资源，该参数最多包含10个key，每个key下面的value最多10个，结构体不能缺失，key不能为空或者空字符串。

参数	是否必选	参数类型	描述
tags_any	否	Array of **Tags** objects	返回结果包含该参数中任意一个标签对应的资源，该参数最多包含10个key，每个key下面的value最多10个，结构体不能缺失，key不能为空或者空字符串。Key不能重复，同一个key中values不能重复。
not_tags	否	Array of **Tags** objects	返回结果不包含该参数中所有标签对应的资源，该参数最多包含10个key，每个key下面的value最多10个，结构体不能缺失，key不能为空或者空字符串。Key不能重复，同一个key中values不能重复。
not_tags_any	否	Array of **Tags** objects	返回结果不包含该参数中任意一个标签对应的资源，该参数最多包含10个key，每个key下面的value最多10个，结构体不能缺失，key不能为空或者空字符串。Key不能重复，同一个key中values不能重复。
matches	否	String	搜索字段，key为要匹配的字段，当前仅支持resource_name。value为匹配的值。此字段为固定字典值

表 5-150 Tags

参数	是否必选	参数类型	描述
key	否	String	键。 • 不能为空。 • 对于同一资源键值唯一。 • 字符集：A-Z，a-z ， 0-9，‘-’，‘_’，UNICODE字符（\u4E00-\u9FFF）。
values	否	Array of strings	标签值列表。 如果values为空列表，则表示any_value。value之间为或的关系。

响应参数

状态码：200

表 5-151 响应 Body 参数

参数	参数类型	描述
action	String	操作标识(仅限于filter，count) • filter：分页查询 • count：查询总条数，只需按照条件将总条数返回即可
limit	String	查询记录数(action为count时无此参数)如果action为filter默认为1000，limit最多为1000，不能为负数，最小值为1
offset	String	索引位置, 从offset指定的下一条数据开始查询。 查询第一页数据时，不需要传入此参数，查询后续页码数据时，将查询前一页数据时响应体中的值带入此参数(action为count时无此参数)如果action为filter默认为0，必须为数字，不能为负数
tags	Array of **Tags** objects	返回结果包含该参数中所有标签对应的资源，该参数最多包含10个key，每个key下面的value最多10个，结构体不能缺失，key不能为空或者空字符串。
tags_any	Array of **Tags** objects	返回结果包含该参数中任意一个标签对应的资源，该参数最多包含10个key，每个key下面的value最多10个，结构体不能缺失，key不能为空或者空字符串。Key不能重复，同一个key中values不能重复。
not_tags	Array of **Tags** objects	返回结果不包含该参数中所有标签对应的资源，该参数最多包含10个key，每个key下面的value最多10个, 结构体不能缺失，key不能为空或者空字符串。Key不能重复，同一个key中values不能重复。
not_tags_any	Array of **Tags** objects	返回结果不包含该参数中任意一个标签对应的资源，该参数最多包含10个key，每个key下面的value最多10个，结构体不能缺失，key不能为空或者空字符串。Key不能重复，同一个key中values不能重复。
matches	String	搜索字段，key为要匹配的字段，当前仅支持resource_name。value为匹配的值。此字段为固定字典值

表 5-152 Tags

参数	参数类型	描述
key	String	键。 ● 不能为空。 ● 对于同一资源键值唯一。 ● 字符集：A-Z，a-z ， 0-9，‘-’，‘_’，UNICODE字符（\u4E00-\u9FFF）。
values	Array of strings	标签值列表。 如果values为空列表，则表示any_value。value之间为或的关系。

请求示例

- 使用标签过滤资源（通道等），查询记录总数。

```
POST https://{Endpoint}/v2/{project_id}/stream/resource_instances/action

{
  "action" : "count",
  "tags" : [ {
    "key" : "key1",
    "values" : [ "value1", "value2" ]
  }, {
    "key" : "key2",
    "values" : [ "value1", "value2" ]
  } ],
  "matches" : [ {
    "key" : "resource_name",
    "value" : "resource1"
  } ]
}
```

- 使用标签过滤资源（通道等），分页查询。

```
POST https://{Endpoint}/v2/{project_id}/stream/resource_instances/action

{
  "offset" : "0",
  "limit" : "100",
  "action" : "filter",
  "matches" : [ {
    "key" : "resource_name",
    "value" : "resource1"
  } ],
  "tags" : [ {
    "key" : "key1",
    "values" : [ "*value1", "value2" ]
  } ]
}
```

响应示例

状态码：200

使用标签过滤资源（通道等）请求体。

```
{
  "resources" : [ {
```

```
  "resource_detail" : null,
  "resource_id" : "cdfs_cefs_wesas_12_dsad",
  "resource_name" : "resouece1",
  "tags" : [ {
    "key" : "key1",
    "value" : "value1"
  }, {
    "key" : "key2",
    "value" : "value1"
  } ]
 } ],
 "total_count" : 1000
}
```

SDK 代码示例

SDK代码示例如下。

Java

- 使用标签过滤资源（通道等），查询记录总数。

```
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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;

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

public class ListResourcesByTagsSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        ListResourcesByTagsRequest request = new ListResourcesByTagsRequest();
        ListResourceInstancesReq body = new ListResourceInstancesReq();
        List<String> listTagsValues = new ArrayList<>();
        listTagsValues.add("value1");
        listTagsValues.add("value2");
        List<String> listTagsValues1 = new ArrayList<>();
        listTagsValues1.add("value1");
        listTagsValues1.add("value2");
        List<Tags> listbodyTags = new ArrayList<>();
        listbodyTags.add(
```

```
            new Tags()
                .withKey("key1")
                .withValues(listTagsValues1)
        );
        listbodyTags.add(
            new Tags()
                .withKey("key2")
                .withValues(listTagsValues)
        );
        body.withMatches("[{key=resource_name, value=resource1}]");
        body.withTags(listbodyTags);
        body.withAction(ListResourceInstancesReq.ActionEnum.fromValue("count"));
        request.withBody(body);
        try {
            ListResourcesByTagsResponse response = client.listResourcesByTags(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());
        }
    }
}
```

- 使用标签过滤资源（通道等），分页查询。

```
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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;

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

public class ListResourcesByTagsSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        ListResourcesByTagsRequest request = new ListResourcesByTagsRequest();
        ListResourceInstancesReq body = new ListResourceInstancesReq();
        List<String> listTagsValues = new ArrayList<>();
```

```
        listTagsValues.add("*value1");
        listTagsValues.add("value2");
        List<Tags> listbodyTags = new ArrayList<>();
        listbodyTags.add(
            new Tags()
                .withKey("key1")
                .withValues(listTagsValues)
        );
        body.withMatches("[{key=resource_name, value=resource1}]");
        body.withTags(listbodyTags);
        body.withOffset("0");
        body.withLimit("100");
        body.withAction(ListResourceInstancesReq.ActionEnum.fromValue("filter"));
        request.withBody(body);
        try {
            ListResourcesByTagsResponse response = client.listResourcesByTags(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

- 使用标签过滤资源（通道等），查询记录总数。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListResourcesByTagsRequest()
        listValuesTags = [
            "value1",
            "value2"
        ]
        listValuesTags1 = [
            "value1",
            "value2"
```

```
        ]
        listTagsbody = [
            Tags(
                key="key1",
                values=listValuesTags1
            ),
            Tags(
                key="key2",
                values=listValuesTags
            )
        ]
        request.body = ListResourceInstancesReq(
            matches="[{key=resource_name, value=resource1}]",
            tags=listTagsbody,
            action="count"
        )
        response = client.list_resources_by_tags(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

- 使用标签过滤资源（通道等），分页查询。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListResourcesByTagsRequest()
        listValuesTags = [
            "*value1",
            "value2"
        ]
        listTagsbody = [
            Tags(
                key="key1",
                values=listValuesTags
            )
        ]
        request.body = ListResourceInstancesReq(
            matches="[{key=resource_name, value=resource1}]",
            tags=listTagsbody,
            offset="0",
            limit="100",
            action="filter"
        )
```

```
        response = client.list_resources_by_tags(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

- 使用标签过滤资源（通道等），查询记录总数。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListResourcesByTagsRequest{}
    var listValuesTags = []string{
        "value1",
        "value2",
    }
    var listValuesTags1 = []string{
        "value1",
        "value2",
    }
    keyTags:= "key1"
    keyTags1:= "key2"
    var listTagsbody = []model.Tags{
        {
            Key: &keyTags,
            Values: &listValuesTags1,
        },
        {
            Key: &keyTags1,
            Values: &listValuesTags,
        },
    }
    matchesListResourceInstancesReq:= "[{key=resource_name, value=resource1}]"
    request.Body = &model.ListResourceInstancesReq{
        Matches: &matchesListResourceInstancesReq,
        Tags: &listTagsbody,
```

```
        Action: model.GetListResourceInstancesReqActionEnum().COUNT,
    }
    response, err := client.ListResourcesByTags(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

- 使用标签过滤资源（通道等），分页查询。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListResourcesByTagsRequest{}
    var listValuesTags = []string{
        "*value1",
        "value2",
    }
    keyTags:= "key1"
    var listTagsbody = []model.Tags{
        {
            Key: &keyTags,
            Values: &listValuesTags,
        },
    }
    matchesListResourceInstancesReq:= "[{key=resource_name, value=resource1}]"
    offsetListResourceInstancesReq:= "0"
    limitListResourceInstancesReq:= "100"
    request.Body = &model.ListResourceInstancesReq{
        Matches: &matchesListResourceInstancesReq,
        Tags: &listTagsbody,
        Offset: &offsetListResourceInstancesReq,
        Limit: &limitListResourceInstancesReq,
        Action: model.GetListResourceInstancesReqActionEnum().FILTER,
    }
    response, err := client.ListResourcesByTags(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	使用标签过滤资源（通道等）请求体。

错误码

请参见**错误码**。

5.7.7 批量删除资源标签

功能介绍

该接口用于批量删除资源（通道等）标签。此接口为幂等接口：删除时，如果删除的标签不存在，默认处理成功；删除时不对标签字符集范围做校验。删除时tags结构体不能缺失，key不能为空，或者空字符串。

调用方法

请参见**如何调用API**。

URI

POST /v2/{project_id}/stream/{stream_id}/tags/action

表 5-153 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
stream_id	是	String	通道ID。

请求参数

表 5-154 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 5-155 请求 Body 参数

参数	是否必选	参数类型	描述
action	是	String	操作标识：仅限于delete • delete：批量删除
tags	是	Array of **Tag** objects	标签列表。

表 5-156 Tag

参数	是否必选	参数类型	描述
key	否	String	键。 • 不能为空。 • 对于同一资源键值唯一。 • 字符集：A-Z，a-z ， 0-9，‘-’，‘_’，UNICODE字符（\u4E00-\u9FFF）。
value	否	String	值。 • 长度不超过43个字符。 • 字符集：A-Z，a-z ， 0-9，‘.’，‘-’，‘_’，UNICODE字符（\u4E00-\u9FFF）。 • 只能包含数字、字母、中划线“-”、下划线“_”。

响应参数

无

请求示例

批量删除资源标签

```
POST https://{Endpoint}/v2/{project_id}/stream/{stream_id}/tags/action

{
  "action" : "delete",
  "tags" : [ {
    "key" : "key1",
    "value" : "value1"
  }, {
    "key" : "key2",
    "value" : "value3"
  } ]
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

批量删除资源标签

```
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.dis.v2.region.DisRegion;
import com.huaweicloud.sdk.dis.v2.*;
import com.huaweicloud.sdk.dis.v2.model.*;

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

public class BatchDeleteTagsSolution {

    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);

        DisClient client = DisClient.newBuilder()
                .withCredential(auth)
                .withRegion(DisRegion.valueOf("<YOUR REGION>"))
                .build();
        BatchDeleteTagsRequest request = new BatchDeleteTagsRequest();
        request.withStreamId("{stream_id}");
```

```
        BatchDeleteTagsReq body = new BatchDeleteTagsReq();
        List<Tag> listbodyTags = new ArrayList<>();
        listbodyTags.add(
            new Tag()
                .withKey("key1")
                .withValue("value1")
        );
        listbodyTags.add(
            new Tag()
                .withKey("key2")
                .withValue("value3")
        );
        body.withTags(listbodyTags);
        body.withAction(BatchDeleteTagsReq.ActionEnum.fromValue("delete"));
        request.withBody(body);
        try {
            BatchDeleteTagsResponse response = client.batchDeleteTags(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

批量删除资源标签

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdis.v2.region.dis_region import DisRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdis.v2 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 = DisClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DisRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = BatchDeleteTagsRequest()
        request.stream_id = "{stream_id}"
        listTagsbody = [
            Tag(
                key="key1",
                value="value1"
            ),
```

```
            Tag(
                key="key2",
                value="value3"
            )
        ]
        request.body = BatchDeleteTagsReq(
            tags=listTagsbody,
            action="delete"
        )
        response = client.batch_delete_tags(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

批量删除资源标签

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dis "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dis/v2/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 := dis.NewDisClient(
        dis.DisClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.BatchDeleteTagsRequest{}
    request.StreamId = "{stream_id}"
    keyTags:= "key1"
    valueTags:= "value1"
    keyTags1:= "key2"
    valueTags1:= "value3"
    var listTagsbody = []model.Tag{
        {
            Key: &keyTags,
            Value: &valueTags,
        },
        {
            Key: &keyTags1,
            Value: &valueTags1,
        },
    }
```

```
    request.Body = &model.BatchDeleteTagsReq{
        Tags: listTagsbody,
        Action: model.GetBatchDeleteTagsReqActionEnum().DELETE,
    }
    response, err := client.BatchDeleteTags(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
204	正常返回

错误码

请参见**错误码**。

6 附录

6.1 错误码

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。

状态码	错误码	错误信息	描述	处理措施
400	DIS.4117	Invalid ProjectId. %s	用户传入的projectId无效	请检查传入的projectId是否有效，是否传入了其他project的id。
400	DIS.4200	Invalid request. %s	用户的请求无效	请参考API文档检查请求。
400	DIS.4201	Invalid partition_id. %s	用户传入的partition_id无效	请检查partition_id是否无效。
400	DIS.4202	Empty request.	用户的请求为空	请传入有效的请求。
400	DIS.4203	Invalid monitoring period. %s	查询监控信息的startTime无效	请传入有效的时间戳。
400	DIS.4204	The monitoring period cannot be longer than 7 days.	仅允许查询最近7天内的监控信息	请查询最近7天内的监控信息。
400	DIS.4205	Stream is not running.	通道状态不是运行中	请检查通道状态。

状态码	错误码	错误信息	描述	处理措施
400	DIS.4208	Mrs cluster is invalid. %s	创建MRS转储任务时，传入的MRS集群无效	请检查传入的MRS集群名称和ID，集群状态是否为运行中，以及是否为安全模式的集群。
400	DIS.4209	Invalid metrics label. %s	查询监控信息时，传入的监控指标不合法	请参考API文档检查监控指标并修正。
400	DIS.4215	Invalid cursor type. %s	获取数据游标时，传入的游标类型cursor-type不合法	请参考API文档检查cursor-type字段的范围并修正。
400	DIS.4216	Invalid sequence_number. %s	获取数据游标时，传入的序列号starting-sequence-number不合法	请传入有效的starting-sequence-number。
400	DIS.4217	Invalid partition cursor. %s	从DIS通道下载数据时，传入的数据游标partition-cursor无效	请重新获取partition-cursor并下载数据。
400	DIS.4224	Sequence_number out of range. %s	获取数据游标时，传入的序列号starting-sequence-number不在有效范围	请传入有效的starting-sequence-number。
400	DIS.4225	Expired partition cursor. %s	从DIS通道下载数据时，传入的数据游标partition-cursor过期	请重新获取partition-cursor并下载数据。
400	DIS.4226	A partition iterator error occurred or a record to which the SN corresponds has expired. Try to obtain the partition iterator again.	获取数据时，传入的数据游标partition-cursor对应的序列号starting-sequence-number过期	请重新获取数据游标，并用新游标获取数据。

状态码	错误码	错误信息	描述	处理措施
400	DIS.4300	Request error.	请求体错误	请对照API文档修正请求体。
400	DIS.4301	The stream does not exist. %s	通道不存在	请检查传入的通道是否存在。
400	DIS.4302	Partition does not exist. %s	通道的分区不存在	请检查用户传入的分区ID是否存在。
400	DIS.4303	Exceeded traffic control limit.	超出流控	请扩容通道或降低上传速率。
400	DIS.4305	Too many stream requests.	同一时间内用户请求太多	请降低请求频率并重试。
400	DIS.4306	Bucket does not exist. %s	传入的OBS桶不存在	请检查OBS桶是否存在。
400	DIS.4307	The stream already exists.	指定的通道已经存在	请修改通道名称并重新创建通道
400	DIS.4308	Insufficient quota.	通道或分区的配额不足	请释放配额或提工单修改账号的配额。
400	DIS.4309	Too many request failures. Please try again later.	IP被加入黑名单	由于频繁的错误访问导致用户ip被加入黑名单，请检查认证信息和请求是否有效，并稍后重试。
400	DIS.4310	OBS access error.	访问OBS失败	请检查用户是否有访问OBS的权限。
400	DIS.4319	Partition is expired. %s	分区已过期	该分区已过期失效，主要针对缩容场景，检查该分区是否已失效，使用正确有效的分区。
400	DIS.4329	app quota exceeded.	APP配额超出限制	请释放APP的配额。
400	DIS.4330	app already exist.	已经存在同名的APP	请修改APP名称并重新创建APP。
400	DIS.4331	app is using.	删除app时，当前app在使用中	请确认app是否在使用中，如需删除请停止使用并重新删除。

状态码	错误码	错误信息	描述	处理措施
400	DIS.4332	app not found.	指定的APP不存在	请检查指定的APP名称是否正确
400	DIS.4335	Invalid IAM agency.	创建转储任务时，使用的IAM委托无效	检查DIS创建的dis_admin_agency或用户自定义的IAM委托是否存在，权限是否完整。
400	DIS.4336	Invalid HDFS path.	创建MRS转储任务时，传入的MRS HDFS路径无效	请检查传入的MRS HDFS路径是否存在。
400	DIS.4337	The DLI database does not exist.	创建DLI转储任务时，传入的DLI数据库不存在	请检查传入的DLI数据库是否存在。
400	DIS.4338	The DLI table does not exist.	创建DLI转储任务时，传入的DLI数据表不存在	请检查传入的DLI表是否存在，并且是否为DLI内表。
400	DIS.4339	Consumer quota exceeded.	消费组消费者配额不足	该消费组内的消费者数量已经超越最大配额，请合理分配消费者或者使用新消费组满足诉求。
400	DIS.4341	The CloudTable cluster does not exist.	创建CloudTable转储任务时，传入的CloudTable集群不存在	请检查传入的CloudTable集群是否存在，集群状态是否正常。
400	DIS.4342	The CloudTable table does not exist	创建CloudTable转储任务时，传入的CloudTable表不存在	请检查传入的CloudTable表是否存在。
400	DIS.4343	The CloudTable table family does not exist.	创建CloudTable转储任务时，传入的CloudTable表的列族不存在	请检查传入的CloudTable表的列族名称是否存在。

状态码	错误码	错误信息	描述	处理措施
400	DIS.4345	Invalid CloudTable schema.	创建CloudTable转储任务时，传入的schema无效	请根据返回的详细信息检查schema，例如配置的JSON属性名称是否存在，参数是否合法等。
400	DIS.4348	Invalid CloudTable openTSDB schema.	创建CloudTable openTSDB转储任务时，传入的schema无效	请根据返回的详细信息检查schema，例如配置的JSON属性名称是否存在，参数是否合法等。
400	DIS.4350	Invalid DWS cluster.	创建DWS转储任务时，传入的DWS集群不存在	请检查DWS集群是否存在，运行是否正常。
400	DIS.4351	Invalid KMS userKey.	创建DWS转储任务时，传入的KMS密钥信息无效	请检查KMS密钥是否存在。
400	DIS.4354	The transfer task does not exist.	删除或更新转储任务时，转储任务不存在	请检查转储任务是否存在。
400	DIS.4355	The transfer task already exists.	创建转储任务时，同一个通道下已存在同名的转储任务	请修改新创建转储任务的名称并重新创建。
400	DIS.4357	Exceeded transfer task quota.	单个通道仅允许同时存在5个转储任务，再创建新的转储任务会超出配额限制	请删除废弃的转储任务释放配额。
400	DIS.4360	Invalid data schema.	创建通道或更新通道时，传入的data_schema无效	请检查data_schema的格式并重试。
400	DIS.4375	The app does not commit checkpoint	该app没有在通道中提交checkpoint操作	请确认该app是否已在消费通道提交了checkpoint操作

状态码	错误码	错误信息	描述	处理措施
400	DIS.4601	The number of resource tags has reached the maximum.	一个资源上最多有10个标签，添加标签时资源上已添加的标签数超出限制	请删除废弃的标签并重新添加标签。
400	DIS.4602	Invalid resource type.	资源类型不合法	请检查资源类型是否合法。
400	DIS.4603	The resource does not exist.	资源不存在	请确认该资源是否已被删除。
400	DIS.4604	The key does not exist.	标签Key不存在	请确认标签Key是否存在。
400	DIS.4605	The action is not supported.	当前标签操作不支持	请确认当前标签操作是否合法，当前仅支持create和delete操作。
403	DIS.4116	Invalid RBAC. %s	用户操作受限	请根据返回的具体信息判断账号是否欠费、无DIS服务的操作权限等。
500	DIS.5000	System error.	系统错误，请联系客服或技术支持工程师处理。	系统错误，请联系客服或技术支持工程师处理。

6.2 状态码

状态码是每次API请求响应的HTTPS状态码，表示本次HTTPS请求服务器返回的状态。

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。
200	OK	服务器已成功处理了请求。
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。

状态码	编码	状态说明
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HTTPS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。
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。

状态码	编码	状态说明
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。
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。
441	Authentication Error	鉴权失败。
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。

状态码	编码	状态说明
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。

6.3 获取项目 ID

介绍如何在控制台或者调用API获取项目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。

获取项目ID的接口为“GET https://{Endpoint}/v3/projects”，其中{Endpoint}为IAM的终端节点，可以从**地区和终端节点**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"
    }
}
```

从控制台获取项目 ID

在调用接口的时候，部分URL中需要填入项目编号，所以需要获取到项目编号。项目编号获取步骤如下：

1. 登录管理控制台。

2. 单击用户名，在下拉列表中单击“基本信息”。
3. 在基本信息页面单击“管理我的凭证”。
   在“我的凭证”页面的项目列表中查看项目ID。

图 6-1 查看项目 ID