

数据库安全服务

API 参考

文档版本 01
发布日期 2024-08-01



版权所有 © 华为云计算技术有限公司 2024。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明



HUAWEI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。

华为云计算技术有限公司

地址：贵州省贵安新区黔中大道交兴功路华为云数据中心 邮编：550029

网址：<https://www.huaweicloud.com/>

目录

1 使用前必读	1
1.1 概述	1
1.2 调用说明	1
1.3 终端节点	1
1.4 基本概念	1
2 如何调用 API	3
2.1 构造请求	3
2.2 认证鉴权	5
2.3 返回结果	7
3 API	9
3.1 管理侧接口	9
3.1.1 开启关闭风险规则	9
3.1.2 开启关闭 Agent	14
3.1.3 查询账户配额信息	19
3.1.4 查询 ECS 服务器规格信息	24
3.1.5 查询可用区信息	29
3.1.6 查询用户操作日志信息	34
3.1.7 修改安全组	41
3.2 实例购买查询接口	46
3.2.1 查询审计实例列表	47
3.2.2 查询实例创建任务信息	54
3.2.3 包年包月计费模式创建审计实例	59
3.3 数据库接口	69
3.3.1 查询数据库列表	69
3.3.2 添加 RDS 免 agent 数据库	75
3.4 标签相关接口	82
3.4.1 查询项目标签	82
3.4.2 根据标签查询资源实例列表	87
3.4.3 根据标签查询资源实例数量	99
3.4.4 批量添加资源标签	110
3.4.5 批量删除资源标签	116
3.5 实例侧规则接口	121

3.5.1 查询审计范围策略列表.....	122
3.5.2 查询 SQL 注入规则策略.....	127
3.5.3 查询风险规则策略.....	133
3.5.4 查询指定风险规则策略.....	139
3.5.5 查询隐私数据脱敏规则.....	145
4 附录.....	152
4.1 状态码.....	152
4.2 错误码.....	153
4.3 获取项目 ID.....	154
5 修订记录.....	156

1 使用前必读

1.1 概述

欢迎使用数据库安全服务（Database Security Service, DBSS）。数据库安全服务是一个智能的数据库安全服务，基于大数据分析技术，提供数据库审计，SQL注入攻击检测，风险操作识别等功能，保障云上数据库的安全。

您可以使用本文档提供的API对实例、规则进行相关操作，如创建、查询、删除等。支持的全部操作请参见[API](#)。

在调用数据库安全服务API之前，请确保已经充分了解数据库安全服务相关概念，详细信息请参见[产品介绍](#)。

1.2 调用说明

数据库安全服务提供了REST（Representational State Transfer）风格API，支持您通过HTTPS请求调用，调用方法请参见[如何调用API](#)。

1.3 终端节点

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

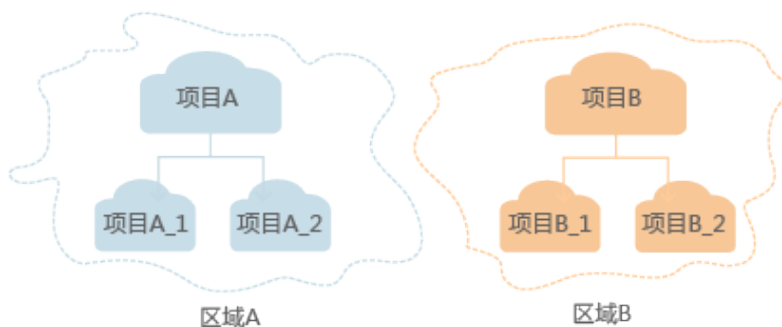
1.4 基本概念

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

- 区域 (Region)
从地理位置和网络时延维度划分，同一个Region内共享弹性计算、块存储、对象存储、VPC网络、弹性公网IP、镜像等公共服务。Region分为通用Region和专属Region，通用Region指面向公共租户提供通用云服务的Region；专属Region指只承载同一类业务或只面向特定租户提供业务服务的专用Region。
- 可用区 (AZ, Availability Zone)
一个AZ是一个或多个物理数据中心的集合，有独立的风火水电，AZ内逻辑上再将计算、网络、存储等资源划分成多个集群。一个Region中的多个AZ间通过高速光纤相连，以满足用户跨AZ构建高可用性系统的需求。
- 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



2 如何调用 API

2.1 构造请求

本节介绍如何构造REST API的请求，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}

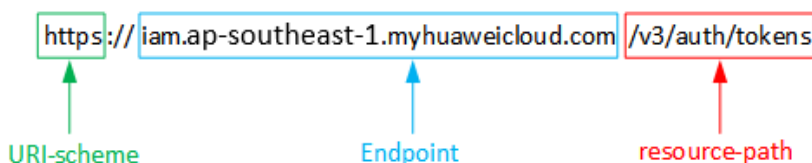
尽管请求URI包含在请求消息头中，但大多数语言或框架都要求您从请求消息中单独传递它，所以在此单独强调。

- **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](#)获取。
例如IAM服务在“中国-香港”区域的Endpoint为“iam.ap-southeast-1.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“中国-香港”区域的Token，则需使用“中国-香港”区域的Endpoint（iam.ap-southeast-1.myhuaweicloud.com），并在[获取用户Token](#)的URI部分找到resource-path（/v3/auth/tokens），拼接起来如下所示。

```
https://iam.ap-southeast-1.myhuaweicloud.com/v3/auth/tokens
```

图 2-1 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，同一个服务的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。

- **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.ap-southeast-1.myhuaweicloud.com/v3/auth/tokens
```

请求消息头

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

如下公共消息头需要添加到请求中。

- **Content-Type**：消息体的类型（格式），必选，默认取值为“application/json”，有其他取值时会在具体接口中专门说明。
- **X-Auth-Token**：用户Token，可选，当使用Token方式认证时，必须填充该字段。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。

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**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.ap-southeast-1.myhuaweicloud.com/v3/auth/tokens
Content-Type: application/json
```


请求消息体

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

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

对于**获取用户Token**接口，您可以从接口的请求部分看到所需的请求参数及参数说明。将消息体加入后的请求如下所示，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，***xxxxxxxxxxxxxxxxxxxx***为project的名称，您可以从**地区和终端节点**获取，对应地区和终端节点页面的“区域”字段的值。

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.ap-southeast-1.myhuaweicloud.com/v3/auth/tokens
Content-Type: application/json
```

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

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。

2.2 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认证调用请求。
- AK/SK认证：通过AK（Access Key ID）/SK（Secret Access Key）加密调用请求。推荐使用AK/SK认证，其安全性比Token认证要高。

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。

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": {
        "name": "xxxxxxx"
      }
    }
  }
}
```

获取Token后，再调用其他接口时，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“ABCDEFJ....”，则调用接口时将“X-Auth-Token: ABCDEFJ....”加到请求消息头即可，如下所示。

```
POST https://iam.ap-southeast-1.myhuaweicloud.com/v3/auth/projects
Content-Type: application/json
X-Auth-Token: ABCDEFJ....
```

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。

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。

2.3 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2-2](#)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 2-2 获取用户 Token 响应消息头

```
connection → keep-alive
content-type → application/json
date → Tue, 12 Feb 2019 06:52:13 GMT
server → Web Server
strict-transport-security → max-age=31536000; includeSubdomains;
transfer-encoding → chunked
via → proxy A
x-content-type-options → nosniff
x-download-options → noopen
x-frame-options → SAMEORIGIN
x-iam-trace-id → Z18d45ab-d674-4995-af3a-2d0255ba41b5
x-subject-token → MIIYXQYJKoZIhvcNAQcCoIIVTjCCGEoCAQExDTALBglghkgBZQMEAgEwgharBgkqhkiG9w0BBwGgghacBIIIWmHsidG9rZW4iOansiZXhwaXJlc19hdCI6IjwMTktMDItMTNUMDfj3Kjs6YgKnpVNRbW2eZ5eb78SZOkajACgkqO1wi4JlGzrpd18LGXK5bdfq4iqHCYb8P4NaY0NYejcAgzJVeFYtLWT1GSO0zxKZmiQHQ82HBqHdglZO9fuEbL5dMhdavj+33wElxHRC9I87o+k9-j+CMZSEB7bUGd5Uj6eRASXl1jipPEGA270g1FruooL6jqglFkNPQuFSOU8+uSsttVwRtnfsC+qTp22Rkd5MCqFGQ8LcuUxC3a+9CM8nOintWW7oeRUVhVpxk8pxiX1wTEboX-RzT6MUbpvGw-oPNFYxJECKnoH3HRozv0vN--n5d6Nbxg==
x-xss-protection → 1; mode=block;
```

响应消息体（可选）

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

对于[获取用户Token](#)接口，返回如下消息体。为篇幅起见，这里只展示部分内容。

```
{
  "token": {
    "expires_at": "2019-02-13T06:52:13.855000Z",
```

```
"methods": [  
  "password"  
],  
"catalog": [  
  {  
    "endpoints": [  
      {  
        "region_id": "xxxxxxx",  
.....
```

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": {  
    "message": "The request you have made requires authentication.",  
    "title": "Unauthorized"  
  }  
}
```

其中，error_code表示错误码，error_msg表示错误描述信息。

3 API

3.1 管理侧接口

3.1.1 开启关闭风险规则

功能介绍

开启关闭风险规则

调用方法

请参见[如何调用API](#)。

URI

POST /v1/{project_id}/{instance_id}/audit/rule/risk/switch

表 3-1 路径参数

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

请求参数

表 3-2 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 3-3 请求 Body 参数

参数	是否必选	参数类型	描述
ids	否	String	risk id, ids 中间逗号分隔
status	否	String	OFF: 关闭 ON: 开启 枚举值: <ul style="list-style-type: none"> • OFF • ON

响应参数

状态码： 200

表 3-4 响应 Body 参数

参数	参数类型	描述
status	String	响应状态

状态码： 400

表 3-5 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-6 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息。

状态码： 403

表 3-7 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-8 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

请求示例

```
/v1/{project_id}/{instance_id}/audit/rule/risk/switch  
{  
  "ids" : "c71LB3kBCwCqSg3B2OpF",  
  "status" : "OFF"  
}
```

响应示例

状态码： 200

请求已成功。

```
{  
  "result" : "SUCCESS"  
}
```

状态码： 400

请求参数有误。

```
{  
  "error" : {  
    "error_code" : "DBSS.XXX",  
    "error_msg" : "XXX"  
  }  
}
```

状态码： 403

认证失败。

```
{  
  "error" : {
```

```
"error_code" : "DBSS.XXX",  
"error_msg" : "XXX"  
}  
}
```

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.dbss.v1.region.DbssRegion;  
import com.huaweicloud.sdk.dbss.v1.*;  
import com.huaweicloud.sdk.dbss.v1.model.*;  
  
public class SwitchRiskRuleSolution {  
  
    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");  
  
        ICredential auth = new BasicCredentials()  
            .withAk(ak)  
            .withSk(sk);  
  
        DbssClient client = DbssClient.newBuilder()  
            .withCredential(auth)  
            .withRegion(DbssRegion.valueOf("cn-north-4"))  
            .build();  
        SwitchRiskRuleRequest request = new SwitchRiskRuleRequest();  
        BatchSwitchesRequest body = new BatchSwitchesRequest();  
        body.withStatus(BatchSwitchesRequest.StatusEnum.fromValue("OFF"));  
        body.withIds("c7ILB3kBCwCqSg3B2OpF");  
        request.withBody(body);  
        try {  
            SwitchRiskRuleResponse response = client.switchRiskRule(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
```



```
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdbss.v1.region.dbss_region import DbssRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdbss.v1 import *

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

    credentials = BasicCredentials(ak, sk) \

    client = DbssClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DbssRegion.value_of("cn-north-4")) \
        .build()

    try:
        request = SwitchRiskRuleRequest()
        request.body = BatchSwitchesRequest(
            status="OFF",
            ids="c71LB3kBCwCqSg3B2OpF"
        )
        response = client.switch_risk_rule(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"
    dbss "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/region"
)

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

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

    client := dbss.NewDbssClient(
        dbss.DbssClientBuilder().
            WithRegion(region.ValueOf("cn-north-4")).
            WithCredential(auth).
            Build())

    request := &model.SwitchRiskRuleRequest{
        statusBatchSwitchesRequest:= model.GetBatchSwitchesRequestStatusEnum().OFF
    }
```

```
idsBatchSwitchesRequest:= "c7ILB3kBCwCqSg3B2OpF"  
request.Body = &model.BatchSwitchesRequest{  
    Status: &statusBatchSwitchesRequest,  
    Ids: &idsBatchSwitchesRequest,  
}  
response, err := client.SwitchRiskRule(request)  
if err == nil {  
    fmt.Printf("%+v\n", response)  
} else {  
    fmt.Println(err)  
}  
}
```

更多

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

状态码

状态码	描述
200	请求已成功。
400	请求参数有误。
403	认证失败。

错误码

请参见[错误码](#)。

3.1.2 开启关闭 Agent

功能介绍

用于开启和关闭agent的功能，当开启后，开始抓取用户的访问信息。

调用方法

请参见[如何调用API](#)。

URI

POST /v1/{project_id}/{instance_id}/audit/agent/switch

表 3-9 路径参数

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

请求参数

表 3-10 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 3-11 请求 Body 参数

参数	是否必选	参数类型	描述
agent_id	是	String	审计agent的ID
status	是	Integer	Agent开关状态 1: 开启 0: 关闭

响应参数

状态码： 200

表 3-12 响应 Body 参数

参数	参数类型	描述
result	String	响应状态

状态码： 400

表 3-13 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-14 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 403

表 3-15 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-16 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

请求示例

```
/v1/{project_id}/{instance_id}/audit/agent/switch  
  
{  
  "agent_id": "ASWDSDSDSWEWDSDS",  
  "status": 1  
}
```

响应示例

状态码： 200

请求已成功。

```
{  
  "result": "SUCCESS"  
}
```

状态码： 400

请求参数有误。

```
{  
  "error": {  
    "error_code": "DBSS.XXX",  
    "error_msg": "XXX"  
  }  
}
```

状态码： 403

认证失败。

```
{  
  "error": {  
    "error_code": "DBSS.XXX",  
    "error_msg": "XXX"  
  }  
}
```

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.dbss.v1.region.DbssRegion;
import com.huaweicloud.sdk.dbss.v1.*;
import com.huaweicloud.sdk.dbss.v1.model.*;

public class SwitchAgentSolution {

    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");

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

        DbssClient client = DbssClient.newBuilder()
            .withCredential(auth)
            .withRegion(DbssRegion.valueOf("cn-north-4"))
            .build();
        SwitchAgentRequest request = new SwitchAgentRequest();
        AgentSwitchRequest body = new AgentSwitchRequest();
        body.withStatus(1);
        body.withAgentId("ASWDSDSDSWEWDSDS");
        request.withBody(body);
        try {
            SwitchAgentResponse response = client.switchAgent(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

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdbss.v1.region.dbss_region import DbssRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdbss.v1 import *
```

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

    credentials = BasicCredentials(ak, sk) \

    client = DbssClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DbssRegion.value_of("cn-north-4")) \
        .build()

    try:
        request = SwitchAgentRequest()
        request.body = AgentSwitchRequest(
            status=1,
            agent_id="ASWDSDSDSWEWDSDS"
        )
        response = client.switch_agent(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"
    dbss "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/region"
)

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

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

    client := dbss.NewDbssClient(
        dbss.DbssClientBuilder().
            WithRegion(region.ValueOf("cn-north-4")).
            WithCredential(auth).
            Build())

    request := &model.SwitchAgentRequest{}
    request.Body = &model.AgentSwitchRequest{
        Status: int32(1),
        AgentId: "ASWDSDSDSWEWDSDS",
    }
    response, err := client.SwitchAgent(request)
    if err == nil {
```

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

更多

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

状态码

状态码	描述
200	请求已成功。
400	请求参数有误。
403	认证失败。

错误码

请参见[错误码](#)。

3.1.3 查询账户配额信息

功能介绍

查询账户配额信息

调用方法

请参见[如何调用API](#)。

URI

GET /v1/{project_id}/dbss/audit/quota

表 3-17 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 3-18 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

响应参数

状态码： 200

表 3-19 响应 Body 参数

参数	参数类型	描述
audit_quota	Long	实例当前剩余配额。
cpu	Long	Cpu当前剩余配额。
project_id	String	项目Id。
quota	Long	配额。
ram	Long	内存当前剩余配额

状态码： 400

表 3-20 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-21 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 403

表 3-22 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-23 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 500

表 3-24 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-25 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

请求示例

```
/v1/{project_id}/dbss/audit/quota
```

响应示例

状态码： 200

成功

```
{  
  "project_id": "0250cb8a80c24c0b9f20f557cb159aad",  
  "cpu": 796,  
  "ram": 1622016,  
  "audit_quota": 1  
}
```

状态码： 400

客户端错误

```
{  
  "error": {
```

```
"error_code" : "DBSS.XXXX",  
"error_msg" : "XXX"  
}  
}
```

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.dbss.v1.region.DbssRegion;  
import com.huaweicloud.sdk.dbss.v1.*;  
import com.huaweicloud.sdk.dbss.v1.model.*;  
  
public class ShowAuditQuotaSolution {  
  
    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");  
  
        ICredential auth = new BasicCredentials()  
            .withAk(ak)  
            .withSk(sk);  
  
        DbssClient client = DbssClient.newBuilder()  
            .withCredential(auth)  
            .withRegion(DbssRegion.valueOf("cn-north-4"))  
            .build();  
        ShowAuditQuotaRequest request = new ShowAuditQuotaRequest();  
        try {  
            ShowAuditQuotaResponse response = client.showAuditQuota(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  
  
from huaweicloudsdkcore.auth.credentials import BasicCredentials  
from huaweicloudsdkdbss.v1.region.dbss_region import DbssRegion  
from huaweicloudsdkcore.exceptions import exceptions
```

```
from huaweicloudsdkdbss.v1 import *

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

    credentials = BasicCredentials(ak, sk) \

    client = DbssClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DbssRegion.value_of("cn-north-4")) \
        .build()

    try:
        request = ShowAuditQuotaRequest()
        response = client.show_audit_quota(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"
    dbss "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/region"
)

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

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

    client := dbss.NewDbssClient(
        dbss.DbssClientBuilder().
            WithRegion(region.ValueOf("cn-north-4")).
            WithCredential(auth).
            Build())

    request := &model.ShowAuditQuotaRequest{}
    response, err := client.ShowAuditQuota(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	成功
400	客户端错误
403	认证失败
500	服务端错误

错误码

请参见[错误码](#)。

3.1.4 查询 ECS 服务器规格信息

功能介绍

查询ECS服务器规格信息

调用方法

请参见[如何调用API](#)。

URI

GET /v1/{project_id}/dbss/audit/specification

表 3-26 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 3-27 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

响应参数

状态码： 200

表 3-28 响应 Body 参数

参数	参数类型	描述
specification	Array of EcsSpecificationBean objects	ecs规格集合

表 3-29 EcsSpecificationBean

参数	参数类型	描述
azs	Array of strings	可用区集合
id	String	ID
level	String	等级
name	String	名称
proxy	Integer	代理
ram	Integer	内存
vcpus	Integer	CPU

状态码： 400

表 3-30 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-31 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 403

表 3-32 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-33 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 500

表 3-34 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-35 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

请求示例

```
/v1/{project_id}/dbss/audit/specification
```

响应示例

状态码： 200

成功

```
{  
  "specification": [ {  
    "level": "low",  
    "id": "s2.xlarge.4",  
    "name": "s2.xlarge.4",  
    "vcpus": 4,  
    "ram": 16384,  
    "proxy": 3,  
    "azs": [ "cn-cmcc1a-01" ]  
  }, {  
    "level": "medium",  
    "id": "s2.2xlarge.4",
```

```
"name" : "s2.2xlarge.4",
"vcpu" : 8,
"ram" : 32768,
"proxy" : 6,
"azs" : [ "cn-cmcc1a-01" ]
}, {
"level" : "high",
"id" : "s3.4xlarge.4",
"name" : "s3.4xlarge.4",
"vcpu" : 16,
"ram" : 65536,
"proxy" : 30,
"azs" : [ "cn-cmcc1a-01", "cn-cmcc1b-01" ]
}]
}
```

状态码： 400

客户端错误

```
{
"error" : {
"error_code" : "DBSS.XXXX",
"error_msg" : "XXX"
}
}
```

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.dbss.v1.region.DbssRegion;
import com.huaweicloud.sdk.dbss.v1.*;
import com.huaweicloud.sdk.dbss.v1.model.*;

public class ListEcsSpecificationSolution {

    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");

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

        DbssClient client = DbssClient.newBuilder()
            .withCredential(auth)
            .withRegion(DbssRegion.valueOf("cn-north-4"))
            .build();
        ListEcsSpecificationRequest request = new ListEcsSpecificationRequest();
        try {
            ListEcsSpecificationResponse response = client.listEcsSpecification(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

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdbss.v1.region.dbss_region import DbssRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdbss.v1 import *

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

    credentials = BasicCredentials(ak, sk) \

    client = DbssClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DbssRegion.value_of("cn-north-4")) \
        .build()

    try:
        request = ListEcsSpecificationRequest()
        response = client.list_ecs_specification(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"
    dbss "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/region"
)

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



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

client := dbss.NewDbssClient(
    dbss.DbssClientBuilder().
        WithRegion(region.ValueOf("cn-north-4")).
        WithCredential(auth).
        Build())

request := &model.ListEcsSpecificationRequest{}
response, err := client.ListEcsSpecification(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	成功
400	客户端错误
403	认证失败
500	服务端错误

错误码

请参见[错误码](#)。

3.1.5 查询可用区信息

功能介绍

查询可用区信息

调用方法

请参见[如何调用API](#)。

URI

GET /v2/{project_id}/dbss/audit/availability-zone

表 3-36 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 3-37 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token

响应参数

状态码： 200

表 3-38 响应 Body 参数

参数	参数类型	描述
azs	Array of AzInfo objects	az列表

表 3-39 AzInfo

参数	参数类型	描述
zone_name	String	可用区名称
zone_number	Integer	可用区编号
az_type	String	可用区类型
alias	String	可用区别名
alias_us	String	可用区别名英文

状态码： 400

表 3-40 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-41 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 403

表 3-42 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-43 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 500

表 3-44 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-45 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

请求示例

```
/v2/{project_id}/dbss/audit/availability-zone
```

响应示例

状态码： 200

成功

```
{
  "azs": [ {
    "zone_name": "cn-north-7b",
    "zone_number": 2,
    "az_type": "normal",
    "alias": "可用区2",
    "alias_us": "AZ2"
  }, {
    "zone_name": "cn-north-7a",
    "zone_number": 1,
    "az_type": "normal",
    "alias": "可用区1",
    "alias_us": "AZ1"
  }, {
    "zone_name": "cn-north-7c",
    "zone_number": 3,
    "az_type": "normal",
    "alias": "可用区3",
    "alias_us": "AZ3"
  } ]
}
```

状态码: 400

客户端错误

```
{
  "error": {
    "error_code": "DBSS.XXXX",
    "error_msg": "XXX"
  }
}
```

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.dbss.v1.region.DbssRegion;
import com.huaweicloud.sdk.dbss.v1.*;
import com.huaweicloud.sdk.dbss.v1.model.*;

public class ListAvailabilityZoneInfosSolution {

    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");

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

```
DbssClient client = DbssClient.newBuilder()
    .withCredential(auth)
    .withRegion(DbssRegion.valueOf("cn-north-4"))
    .build();
ListAvailabilityZoneInfosRequest request = new ListAvailabilityZoneInfosRequest();
try {
    ListAvailabilityZoneInfosResponse response = client.listAvailabilityZoneInfos(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

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdbss.v1.region.dbss_region import DbssRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdbss.v1 import *

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

    credentials = BasicCredentials(ak, sk) \

    client = DbssClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DbssRegion.value_of("cn-north-4")) \
        .build()

    try:
        request = ListAvailabilityZoneInfosRequest()
        response = client.list_availability_zone_infos(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"
    dbss "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/region"
)
```

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

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

    client := dbss.NewDbssClient(
        dbss.DbssClientBuilder().
            WithRegion(region.ValueOf("cn-north-4")).
            WithCredential(auth).
            Build())

    request := &model.ListAvailabilityZoneInfosRequest{}
    response, err := client.ListAvailabilityZoneInfos(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	成功
400	客户端错误
403	认证失败
500	服务端错误

错误码

请参见[错误码](#)。

3.1.6 查询用户操作日志信息

功能介绍

查询用户操作日志信息

调用方法

请参见[如何调用API](#)。

URI

POST /v1/{project_id}/{instance_id}/dbss/audit/operate-log

表 3-46 路径参数

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

请求参数

表 3-47 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

表 3-48 请求 Body 参数

参数	是否必选	参数类型	描述
time	否	TimeRangeBean object	时间区间
user_name	否	String	筛选角色用户获取操作日志
operate_name	否	String	筛选操作对象名称获取操作日志
result	否	String	根据执行结果获取操作日志 [success, fail]
page	否	String	页数
size	否	String	每页条数

表 3-49 TimeRangeBean

参数	是否必选	参数类型	描述
end_time	否	String	开始时间，必须和end_time成对出现。格式必须为yyyy-MM-dd HH:mm:ss。UTC时间
start_time	否	String	结束时间，必须和start_time成对出现。格式必须为yyyy-MM-dd HH:mm:ss。UTC时间

参数	是否必选	参数类型	描述
time_range	否	String	请求查询的时间段，值为： HALF_HOUR HOUR THREE_HOUR TWELVE_HOUR DAY（24小时）WEEK（7天） MONTH（30天）枚举值： □ HALF_HOUR □ HOUR □ THREE_HOUR □ TWELVE_HOUR □ DAY □ WEEK □ MONTH

响应参数

状态码：200

表 3-50 响应 Body 参数

参数	参数类型	描述
total_num	Integer	总数
operate_log	Array of OperateLogInfo objects	操作日志列表

表 3-51 OperateLogInfo

参数	参数类型	描述
id	String	操作日志ID
user	String	操作日志用户
time	String	该条记录发生的时间，格式为时间戳
function	String	该条记录的功能类型
action	String	该条记录的操作类型 □ create：创建 □ update：更新 □ operate：操作（开关） □ delete：删除
name	String	该条记录对应的用户操作对象
description	String	该条记录具体的描述
result	String	该条记录对应用户执行的结果 □ success表示成功 □ fail表示失败

状态码：400

表 3-52 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-53 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 403

表 3-54 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-55 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 500

表 3-56 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-57 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

请求示例

```
/v1/{project_id}/{instance_id}/dbss/audit/operate-log  
  
{  
  "time" : {  
    "time_range" : "HOUR"  
  },  
  "page" : 1,  
  "size" : 10  
}
```

响应示例

状态码： 200

成功

```
{  
  "total_num" : 3,  
  "operate_log" : [ {  
    "id" : "1LJP-HgBCwCqSg3BVuAp",  
    "user" : "hby-test",  
    "time" : "2021-04-22 06:40:52",  
    "function" : "数据库列表",  
    "action" : "删除",  
    "name" : "db01 ",  
    "description" : "删除审计的数据库",  
    "result" : "success"  
  }, {  
    "id" : "07JO-HgBCwCqSg3ByOAD",  
    "user" : "hby-test",  
    "time" : "2021-04-22 06:40:15",  
    "function" : "数据库列表",  
    "action" : "更新",  
    "name" : "db01 ",  
    "description" : "关闭审计客户端",  
    "result" : "success"  
  }, {  
    "id" : "ULKM93gBCwCqSg3BZeD1",  
    "user" : "hby-test",  
    "time" : "2021-04-22 03:07:56",  
    "function" : "数据库列表",  
    "action" : "创建",  
    "name" : "db01",  
    "description" : "创建新的数据库",  
    "result" : "success"  
  } ]  
}
```

状态码： 400

请求参数错误

```
{  
  "error" : {  
    "error_code" : "DBSS.XXXX",  
    "error_msg" : "XXX"  
  }  
}
```

状态码： 500

服务器内部错误

```
{  
  "error" : {  
    "error_code" : "DBSS.XXXX",
```

```
"error_msg" : "XXX"  
}  
}
```

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.dbss.v1.region.DbssRegion;  
import com.huaweicloud.sdk.dbss.v1.*;  
import com.huaweicloud.sdk.dbss.v1.model.*;  
  
public class ListAuditOperateLogsSolution {  
  
    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");  
  
        ICredential auth = new BasicCredentials()  
            .withAk(ak)  
            .withSk(sk);  
  
        DbssClient client = DbssClient.newBuilder()  
            .withCredential(auth)  
            .withRegion(DbssRegion.valueOf("cn-north-4"))  
            .build();  
        ListAuditOperateLogsRequest request = new ListAuditOperateLogsRequest();  
        OperateLogGetRequest body = new OperateLogGetRequest();  
        TimeRangeBean timebody = new TimeRangeBean();  
        timebody.withTimeRange("HOUR");  
        body.withSize("10");  
        body.withPage("1");  
        body.withTime(timebody);  
        request.withBody(body);  
        try {  
            ListAuditOperateLogsResponse response = client.listAuditOperateLogs(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.getStatusCode());  
            System.out.println(e.getRequestId());  
            System.out.println(e.getErrorCode());  
            System.out.println(e.getErrorMsg());  
        }  
    }  
}
```

Python

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdbss.v1.region.dbss_region import DbssRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdbss.v1 import *

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

    credentials = BasicCredentials(ak, sk) \

    client = DbssClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DbssRegion.value_of("cn-north-4")) \
        .build()

    try:
        request = ListAuditOperateLogsRequest()
        timebody = TimeRangeBean(
            time_range="HOURL"
        )
        request.body = OperateLogGetRequest(
            size="10",
            page="1",
            time=timebody
        )
        response = client.list_audit_operate_logs(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"
    dbss "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/region"
)

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

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

```
client := dbss.NewDbssClient(
    dbss.DbssClientBuilder().
        WithRegion(region.ValueOf("cn-north-4")).
        WithCredential(auth).
        Build())

request := &model.ListAuditOperateLogsRequest{
    timeRangeTime:= "HOUR"
    timebody := &model.TimeRangeBean{
        TimeRange: &timeRangeTime,
    }
    sizeOperateLogGetRequest:= "10"
    pageOperateLogGetRequest:= "1"
    request.Body = &model.OperateLogGetRequest{
        Size: &sizeOperateLogGetRequest,
        Page: &pageOperateLogGetRequest,
        Time: timebody,
    }
}
response, err := client.ListAuditOperateLogs(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	成功
400	请求参数错误
403	认证失败
500	服务器内部错误

错误码

请参见[错误码](#)。

3.1.7 修改安全组

功能介绍

修改安全组

调用方法

请参见[如何调用API](#)。

URI

POST /v1/{project_id}/dbss/audit/security-group

表 3-58 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 3-59 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

表 3-60 请求 Body 参数

参数	是否必选	参数类型	描述
resource_id	是	String	资源ID
securitygroup_ids	是	Array of strings	安全组ID列表(目前只支持传一个ID)

响应参数

状态码： 200

表 3-61 响应 Body 参数

参数	参数类型	描述
result	String	响应状态

状态码： 400

表 3-62 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-63 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 403

表 3-64 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-65 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 500

表 3-66 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-67 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

请求示例

```
/v1/{project_id}/dbss/audit/security-group  
  
{  
  "resource_id": "062212d8-8e30-4783-9671-43f3f1f3bb1e",  
  "securitygroup_ids": [ "f0fbec06-bcf6-4c7e-99fa-f0ddfb1d9bd" ]  
}
```

响应示例

状态码： 400

失败

```
{
  "error" : {
    "error_code" : "DBSS.XXXX",
    "error_msg" : "XXX"
  }
}
```

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.dbss.v1.region.DbssRegion;
import com.huaweicloud.sdk.dbss.v1.*;
import com.huaweicloud.sdk.dbss.v1.model.*;

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

public class UpdateAuditSecurityGroupSolution {

    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");

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

        DbssClient client = DbssClient.newBuilder()
            .withCredential(auth)
            .withRegion(DbssRegion.valueOf("cn-north-4"))
            .build();
        UpdateAuditSecurityGroupRequest request = new UpdateAuditSecurityGroupRequest();
        SecurityGroupRequest body = new SecurityGroupRequest();
        List<String> listbodySecuritygroupIds = new ArrayList<>();
        listbodySecuritygroupIds.add("f0fbec06-bcf6-4c7e-99fa-f0ddfbb1d9bd");
        body.withSecuritygroupIds(listbodySecuritygroupIds);
        body.withResourceId("062212d8-8e30-4783-9671-43f3f1f3bb1e");
        request.withBody(body);
        try {
            UpdateAuditSecurityGroupResponse response = client.updateAuditSecurityGroup(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
        }
    }
}
```



```
e.printStackTrace();
System.out.println(e.getStatusCode());
System.out.println(e.getRequestId());
System.out.println(e.getErrorCode());
System.out.println(e.getErrorMsg());
    }
}
}
```

Python

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdbss.v1.region.dbss_region import DbssRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdbss.v1 import *

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

    credentials = BasicCredentials(ak, sk) \

    client = DbssClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DbssRegion.value_of("cn-north-4")) \
        .build()

    try:
        request = UpdateAuditSecurityGroupRequest()
        listSecuritygroupIdsbody = [
            "f0fbec06-bcf6-4c7e-99fa-f0ddfbb1d9bd"
        ]
        request.body = SecurityGroupRequest(
            securitygroup_ids=listSecuritygroupIdsbody,
            resource_id="062212d8-8e30-4783-9671-43f3f1f3bb1e"
        )
        response = client.update_audit_security_group(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"
    dbss "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/region"
)

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

```
ak := os.Getenv("CLOUD_SDK_AK")
sk := os.Getenv("CLOUD_SDK_SK")

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

client := dbss.NewDbssClient(
    dbss.DbssClientBuilder().
        WithRegion(region.ValueOf("cn-north-4")).
        WithCredential(auth).
        Build())

request := &model.UpdateAuditSecurityGroupRequest{}
var listSecuritygroupIdsbody = []string{
    "f0fbec06-bcf6-4c7e-99fa-f0ddfb1d9bd",
}
request.Body = &model.SecurityGroupRequest{
    SecuritygroupIds: listSecuritygroupIdsbody,
    ResourceId: "062212d8-8e30-4783-9671-43f3f1f3bb1e",
}
response, err := client.UpdateAuditSecurityGroup(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	成功
400	失败
403	认证失败
500	服务端错误

错误码

请参见[错误码](#)。

3.2 实例购买查询接口

3.2.1 查询审计实例列表

功能介绍

查询审计实例列表

调用方法

请参见[如何调用API](#)。

URI

GET /v1/{project_id}/dbss/audit/instances

表 3-68 路径参数

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

表 3-69 Query 参数

参数	是否必选	参数类型	描述
offset	否	String	偏移量
limit	否	String	查询记录数

请求参数

表 3-70 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

响应参数

状态码： 200

表 3-71 响应 Body 参数

参数	参数类型	描述
servers	Array of AuditInstanceListBean objects	实例信息列表
total	Integer	总数

表 3-72 AuditInstanceListBean

参数	参数类型	描述
charge_model	String	付费模式 Period: 包周期 Demand: 按需。
comment	String	备注信息。
config_num	Integer	配置的数据库总数。
connect_ip	String	连接地址。
connect_ipv6	String	ipv6连接地址。
cpu	Integer	CPU个数
created	String	创建时间
database_limit	Integer	支持的数据库总数
effect	Integer	1.冻结可释放 2.冻结不可释放 3.冻结后不可续费
expired	String	过期时间
id	String	ID
keep_days	String	剩余天数
name	String	实例别名
new_version	String	如果有返回,则需要升级,如果没有,则为null
port_id	String	绑定弹性IP的portID
ram	Integer	内存
region	String	实例所在region
remain_days	String	到期天数
resource_id	String	资源ID
resource_spec_code	String	实例的规格
scene	String	场景

参数	参数类型	描述
security_group_id	String	安全组
specification	String	实例规格
status	String	实例状态： SHUTOFF(已关闭) ACTIVE(运行中，允许任何操作) DELETING(删除中，不允许任何操作) BUILD(创建中，不允许任何操作) DELETED(已删除，不需要展示) ERROR(故障，只允许删除) HAWAIT(等待备机创建成功，不允许任何操作) FROZEN(已冻结，只允许续费、绑定/解绑) UPGRADING(升级中，不允许升级操作)
subnet_id	String	子网ID
task	String	任务状态： powering-on(正在开启，实例可以绑定、解绑) powering-off(正在关闭，实例可以绑定、解绑) rebooting(正在重启，实例可以绑定、解绑) delete_wait(等待删除，集群与实例不允许任何操作) NO_TASK(不展示)
version	String	实例的当前版本
vpc_id	String	虚拟私有云
zone	String	可用区

状态码： 400

表 3-73 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-74 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 403

表 3-75 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-76 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 500

表 3-77 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-78 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

请求示例

```
/v1/{project_id}/dbss/audit/instances
```

响应示例

状态码： 200

成功

```
{  
  "servers": [ {  
    "name": "DBSS-Test",  
    "comment": "",  
    "connect_ipv6": null,  
    "status": "ACTIVE",  
    "task": "NO_TASK",  
    "id": "8c53ed03-8ed7-4ff2-ad97-7b2d6d1dd364",  
    "specification": "Low | 3 Proxy",  
    "zone": "cn-cmcc1a-01",  
    "created": "2021-04-21 04:37:54",  
    "expired": null,  
  }  
]
```

```
"subnet_id" : "97ef0bb5-3759-4db4-aa49-0d087ed49ce5",
"cpu" : 4,
"ram" : 16384,
"region" : "cn-cmcc1",
"version" : "21.04.16.164614",
"charge_model" : "Demand",
"remain_days" : null,
"config_num" : 1,
"effect" : null,
"scene" : null,
"connect_ip" : "192.168.0.229",
"port_id" : "dc4bd420-e01c-4d12-a7ff-814f17c63079",
"resource_id" : "062212d8-8e30-4783-9671-43f3f1f3bb1e",
"vpc_id" : "76d98391-5abc-46ed-b8a8-f664202cb166",
"security_group_id" : "f0fbec06-bcf6-4c7e-99fa-f0ddfbb1d9bd",
"resource_spec_code" : "dbss.bypassaudit.low",
"keep_days" : null,
"new_version" : null,
"database_limit" : 3
}],
"total" : 1
}
```

状态码： 400

失败

```
{
  "error" : {
    "error_code" : "DBSS.XXXX",
    "error_msg" : "XXX"
  }
}
```

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.dbss.v1.region.DbssRegion;
import com.huaweicloud.sdk.dbss.v1.*;
import com.huaweicloud.sdk.dbss.v1.model.*;

public class ListAuditInstancesSolution {

    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");

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

        DbssClient client = DbssClient.newBuilder()
```

```
        .withCredential(auth)
        .withRegion(DbssRegion.valueOf("cn-north-4"))
        .build();
ListAuditInstancesRequest request = new ListAuditInstancesRequest();
request.withOffset("<offset>");
request.withLimit("<limit>");
try {
    ListAuditInstancesResponse response = client.listAuditInstances(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

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdbss.v1.region.dbss_region import DbssRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdbss.v1 import *

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

    credentials = BasicCredentials(ak, sk) \

    client = DbssClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DbssRegion.value_of("cn-north-4")) \
        .build()

    try:
        request = ListAuditInstancesRequest()
        request.offset = "<offset>"
        request.limit = "<limit>"
        response = client.list_audit_instances(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"
    dbss "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/model"
```



```

region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/region"
)

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

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

    client := dbss.NewDbssClient(
        dbss.DbssClientBuilder().
            WithRegion(region.ValueOf("cn-north-4")).
            WithCredential(auth).
            Build())

    request := &model.ListAuditInstancesRequest{}
    offsetRequest:= "<offset>"
    request.Offset = &offsetRequest
    limitRequest:= "<limit>"
    request.Limit = &limitRequest
    response, err := client.ListAuditInstances(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	成功
400	失败
403	认证失败
500	服务端错误

错误码

请参见[错误码](#)。

3.2.2 查询实例创建任务信息

功能介绍

查询实例创建任务信息

调用方法

请参见[如何调用API](#)。

URI

GET /v1/{project_id}/dbss/audit/jobs/{resource_id}

表 3-79 路径参数

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

请求参数

表 3-80 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

响应参数

状态码： 200

表 3-81 响应 Body 参数

参数	参数类型	描述
jobs	Array of JobBean objects	实例创建任务列表

表 3-82 JobBean

参数	参数类型	描述
job_id	String	任务ID,异步查询标识

参数	参数类型	描述
status	String	状态
job_type	String	类型
server_id	String	虚拟机ID
server_name	String	虚拟机名称
begin_time	Long	开始时间
end_time	Long	结束时间
charge_mode	String	计费模式
error_code	String	错误码
fail_reason	String	失败原因
ha_id	String	双机实例HA共用的id
ha_name	String	HA别名

状态码： 400

表 3-83 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-84 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 403

表 3-85 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-86 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 500

表 3-87 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-88 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

请求示例

```
/v1/{project_id}/dbss/audit/jobs/{resource_id}
```

响应示例

状态码： 200

成功

```
{
  "jobs": [ {
    "ha_id": "2c154fdd-0d43-47b7-9cf1-5236bf6a2ca7",
    "ha_name": null,
    "status": "SUCCESS",
    "job_type": null,
    "job_id": "8abf9647852a1daa01852e517e1a1a0b",
    "begin_time": 1671519371000,
    "end_time": 1671519417000,
    "error_code": null,
    "fail_reason": null,
    "charge_mode": "Demand",
    "server_name": "DBSS-qct-1220",
    "server_id": "0aa8f621-bc19-4822-b66d-7ab9ae3c8693"
  } ]
}
```

状态码： 400

失败

```
{
  "error": {
```

```
"error_code" : "DBSS.XXXX",  
"error_msg" : "XXX"  
}  
}
```

状态码： 500

服务器内部错误

```
{  
  "error" : {  
    "error_code" : "DBSS.XXXX",  
    "error_msg" : "XXX"  
  }  
}
```

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.dbss.v1.region.DbssRegion;  
import com.huaweicloud.sdk.dbss.v1.*;  
import com.huaweicloud.sdk.dbss.v1.model.*;  
  
public class ListAuditInstanceJobsSolution {  
  
    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");  
  
        ICredential auth = new BasicCredentials()  
            .withAk(ak)  
            .withSk(sk);  
  
        DbssClient client = DbssClient.newBuilder()  
            .withCredential(auth)  
            .withRegion(DbssRegion.valueOf("cn-north-4"))  
            .build();  
        ListAuditInstanceJobsRequest request = new ListAuditInstanceJobsRequest();  
        try {  
            ListAuditInstanceJobsResponse response = client.listAuditInstanceJobs(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.getStatusCode());  
            System.out.println(e.getRequestId());  
            System.out.println(e.getErrorCode());  
            System.out.println(e.getErrorMsg());  
        }  
    }  
}
```

```
}  
}
```

Python

```
# coding: utf-8  
  
from huaweicloudsdkcore.auth.credentials import BasicCredentials  
from huaweicloudsdkdbss.v1.region.dbss_region import DbssRegion  
from huaweicloudsdkcore.exceptions import exceptions  
from huaweicloudsdkdbss.v1 import *  
  
if __name__ == "__main__":  
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security  
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment  
    # variables and decrypted during use to ensure security.  
    # In this example, AK and SK are stored in environment variables for authentication. Before running this  
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment  
    ak = os.getenv("CLOUD_SDK_AK")  
    sk = os.getenv("CLOUD_SDK_SK")  
  
    credentials = BasicCredentials(ak, sk) \  
  
    client = DbssClient.new_builder() \  
        .with_credentials(credentials) \  
        .with_region(DbssRegion.value_of("cn-north-4")) \  
        .build()  
  
    try:  
        request = ListAuditInstanceJobsRequest()  
        response = client.list_audit_instance_jobs(request)  
        print(response)  
    except exceptions.ClientRequestException as e:  
        print(e.status_code)  
        print(e.request_id)  
        print(e.error_code)  
        print(e.error_msg)
```

Go

```
package main  
  
import (  
    "fmt"  
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"  
    dbss "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1"  
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/model"  
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/region"  
)  
  
func main() {  
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security  
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment  
    // variables and decrypted during use to ensure security.  
    // In this example, AK and SK are stored in environment variables for authentication. Before running this  
    // example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment  
    ak := os.Getenv("CLOUD_SDK_AK")  
    sk := os.Getenv("CLOUD_SDK_SK")  
  
    auth := basic.NewCredentialsBuilder().  
        WithAk(ak).  
        WithSk(sk).  
        Build()  
  
    client := dbss.NewDbssClient(  
        dbss.DbssClientBuilder().  
            WithRegion(region.ValueOf("cn-north-4")).  
            WithCredential(auth).  
            Build())
```

```
request := &model.ListAuditInstanceJobsRequest{}
response, err := client.ListAuditInstanceJobs(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	成功
400	失败
403	认证失败
500	服务器内部错误

错误码

请参见[错误码](#)。

3.2.3 包年包月计费模式创建审计实例

功能介绍

包年包月计费模式创建审计实例

调用方法

请参见[如何调用API](#)。

URI

POST /v2/{project_id}/dbss/audit/charge/period/order

表 3-89 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 3-90 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

表 3-91 请求 Body 参数

参数	是否必选	参数类型	描述
flavor_ref	是	String	云服务器使用的规格ID
name	是	String	云服务器名称。取值范围：只能由中文字符、英文字母、数字、下划线、中划线组成，且长度小于等于64个字符。创建的云服务器数量大于1时，长度小于等于59个字符
vpc_id	是	String	VPC的ID
availability_zone	是	String	云服务器对应可用分区信息。（两个主备分区，中间用“,”分割，例如az1.dc1,az2.dc2）
enterprise_project_id	是	String	企业项目ID
nics	是	Array of nics objects	云服务器对应的网卡信息
security_groups	是	Array of security_groups objects	云服务器对应安全组信息
comment	否	String	备注信息
region	是	String	云服务器所在区域ID
cloud_service_type	是	String	服务类型：默认 hws.service.type.dbss
charging_mode	是	Integer	计费模式：0：包周期计费 1：按需计费
period_type	是	Integer	订购周期类型：0：天；1：周；2：月；3：年；4：小时；5：绝对时间
period_num	是	Integer	订购周期数
subscription_num	是	Integer	订购数量：DBSS只支持订购1套，不支持多套

参数	是否必选	参数类型	描述
product_infos	是	Array of product_infos objects	产品信息列表
tags	否	Array of KeyValueBean objects	资源标签
promotion_info	否	String	折扣信息
is_auto_renew	否	Integer	自动续费 1表示自动续费，0表示不自动续费

表 3-92 nics

参数	是否必选	参数类型	描述
subnet_id	是	String	网卡对应的子网ID
ip_address	否	String	IP地址，不填或空字符串为自动分配

表 3-93 security_groups

参数	是否必选	参数类型	描述
id	是	String	云服务器对应的安全组ID，会对创建云服务器中配置的网卡生效

表 3-94 product_infos

参数	是否必选	参数类型	描述
product_id	是	String	产品ID
cloud_service_type	是	String	服务类型：默认 hws.service.type.dbss
resource_type	是	String	资源类型：默认 hws.resource.type.dbss
resource_spec_code	是	String	资源规格： dbss.bypassaudit.low、 dbss.bypassaudit.medium、 dbss.bypassaudit.high

参数	是否必选	参数类型	描述
product_spec_desc	是	String	产品规格描述， json字符串格式： {"specDesc":{"zh-cn":{"key1":"value1"},"en-us":{"key1":"value1"}}}

表 3-95 KeyValueBean

参数	是否必选	参数类型	描述
key	是	String	键
value	是	String	值

响应参数

状态码： 200

表 3-96 响应 Body 参数

参数	参数类型	描述
description	String	描述
code	String	返回码
order_id	String	订单ID

状态码： 400

表 3-97 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-98 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 403

表 3-99 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-100 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 500

表 3-101 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-102 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

请求示例

创建一个包周期计费"chargin_mode : 0"的实例，版本为专业版：
"resource_spec_code" : "dbss.bypassaudit.medium"。

```
{
  "flavor_ref": "c6.2xlarge.2",
  "name": "DBSS-test",
  "vpc_id": "4c035747-f77b-4c6d-b23b-cb3a2b96c7e6",
  "availability_zone": "cn-north-7b",
  "comment": "",
  "region": "cn-north-7",
  "nics": [
    {
      "subnet_id": "6201dcf2-1374-43ec-ae8b-78b4081572d3"
    }
  ],
  "security_groups": [
    {
      "id": "59c45017-a484-481b-8440-18c2214ccb06"
    }
  ],
  "cloud_service_type": "hws.service.type.dbss",
}
```

```

"charging_mode": 0,
"period_type": 2,
"period_num": 1,
"subscription_num": 1,
"is_auto_renew": 0,
"product_infos": [
  {
    "product_id": "00301-225396-0--0",
    "cloud_service_type": "hws.service.type.dbss",
    "resource_type": "hws.resource.type.dbss",
    "resource_spec_code": "dbss.bypassaudit.medium",
    "product_spec_desc": "{\"specDesc\":{\"zh-cn\":{\"key1\":\"value1\"},\"en-us\":
    {\"key1\":\"value1\"}}}"
  }
],
"promotion_info": "",
"enterprise_project_id": "0"
}

```

响应示例

状态码： 200

成功

```

{
  "description" : "Success",
  "code" : "0",
  "order_id" : "CS1710190909OGQIS"
}

```

状态码： 400

失败

```

{
  "error" : {
    "error_code" : "DBSS.XXXX",
    "error_msg" : "XXX"
  }
}

```

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.dbss.v1.region.DbssRegion;
import com.huaweicloud.sdk.dbss.v1.*;
import com.huaweicloud.sdk.dbss.v1.model.*;

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

public class CreateInstancesPeriodOrderSolution {

    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");

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

DbssClient client = DbssClient.newBuilder()
    .withCredential(auth)
    .withRegion(DbssRegion.valueOf("cn-north-4"))
    .build();
CreateInstancesPeriodOrderRequest request = new CreateInstancesPeriodOrderRequest();
CreateInstancePeriodRequest body = new CreateInstancePeriodRequest();
List<KeyValueBean> listbodyTags = new ArrayList<>();
listbodyTags.add(
    new KeyValueBean()
        .withKey("key_test")
        .withValue("1")
);
List<CreateInstancePeriodRequestProductInfos> listbodyProductInfos = new ArrayList<>();
listbodyProductInfos.add(
    new CreateInstancePeriodRequestProductInfos()
        .withProductId("00301-225396-0--0")
        .withCloudServiceType("hws.service.type.dbss")
        .withResourceType("hws.resource.type.dbss")
        .withResourceSpecCode("dbss.bypassaudit.low")
        .withProductSpecDesc("test")
);
List<CreateInstancePeriodRequestSecurityGroups> listbodySecurityGroups = new ArrayList<>();
listbodySecurityGroups.add(
    new CreateInstancePeriodRequestSecurityGroups()
        .withId("04088976-9c63-4e6b-9070-84e6a30c782b")
);
List<CreateInstancePeriodRequestNics> listbodyNics = new ArrayList<>();
listbodyNics.add(
    new CreateInstancePeriodRequestNics()
        .withSubnetId("6201dcf2-1374-43ec-ae8b-78b4081572d3")
);
body.withIsAutoRenew(0);
body.withPromotionInfo("");
body.withTags(listbodyTags);
body.withProductInfos(listbodyProductInfos);
body.withSubscriptionNum(1);
body.withPeriodNum(1);
body.withPeriodType(2);
body.withChargingMode(0);
body.withCloudServiceType("hws.service.type.dbss");
body.withRegion("cn-north-7");
body.withComment("");
body.withSecurityGroups(listbodySecurityGroups);
body.withNics(listbodyNics);
body.withEnterpriseProjectId("0");
body.withAvailabilityZone("cn-north-7c");
body.withVpcId("4c035747-f77b-4c6d-b23b-cb3a2b96c7e6");
body.withName("DBSS-acc3");
body.withFlavorRef("st6.xlarge.4");
request.withBody(body);
try {
    CreateInstancesPeriodOrderResponse response = client.createInstancesPeriodOrder(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.getStatusCode());
        System.out.println(e.getRequestId());
        System.out.println(e.getErrorCode());
        System.out.println(e.getErrorMsg());
    }
}
}
```

Python

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdbss.v1.region.dbss_region import DbssRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdbss.v1 import *

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

    credentials = BasicCredentials(ak, sk) \

    client = DbssClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DbssRegion.value_of("cn-north-4")) \
        .build()

    try:
        request = CreateInstancesPeriodOrderRequest()
        listTagsbody = [
            KeyValueBean(
                key="key_test",
                value="1"
            )
        ]
        listProductInfosbody = [
            CreateInstancePeriodRequestProductInfos(
                product_id="00301-225396-0--0",
                cloud_service_type="hws.service.type.dbss",
                resource_type="hws.resource.type.dbss",
                resource_spec_code="dbss.bypassaudit.low",
                product_spec_desc="test"
            )
        ]
        listSecurityGroupsbody = [
            CreateInstancePeriodRequestSecurityGroups(
                id="04088976-9c63-4e6b-9070-84e6a30c782b"
            )
        ]
        listNicsbody = [
            CreateInstancePeriodRequestNics(
                subnet_id="6201dcf2-1374-43ec-ae8b-78b4081572d3"
            )
        ]
        request.body = CreateInstancePeriodRequest(
            is_auto_renew=0,
            promotion_info="",
            tags=listTagsbody,
            product_infos=listProductInfosbody,
            subscription_num=1,
            period_num=1,
            period_type=2,
            charging_mode=0,
```

```
cloud_service_type="hws.service.type.dbss",
region="cn-north-7",
comment="",
security_groups=listSecurityGroupsbody,
nics=listNicsbody,
enterprise_project_id="0",
availability_zone="cn-north-7c",
vpc_id="4c035747-f77b-4c6d-b23b-cb3a2b96c7e6",
name="DBSS-acc3",
flavor_ref="st6.xlarge.4"
)
response = client.create_instances_period_order(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"
    dbss "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/region"
)

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

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

    client := dbss.NewDbssClient(
        dbss.DbssClientBuilder().
            WithRegion(region.ValueOf("cn-north-4")).
            WithCredential(auth).
            Build())

    request := &model.CreateInstancesPeriodOrderRequest{}
    var listTagsbody = []model.KeyValueBean{
        {
            Key: "key_test",
            Value: "1",
        },
    }
    var listProductInfosbody = []model.CreateInstancePeriodRequestProductInfos{
        {
            ProductId: "00301-225396-0--0",
            CloudServiceType: "hws.service.type.dbss",
            ResourceType: "hws.resource.type.dbss",
            ResourceSpecCode: "dbss.bypassaudit.low",
            ProductSpecDesc: "test",
        },
    }
    var listSecurityGroupsbody = []model.CreateInstancePeriodRequestSecurityGroups{
        {
```

```

        Id: "04088976-9c63-4e6b-9070-84e6a30c782b",
    },
}
var listNicsbody = []model.CreateInstancePeriodRequestNics{
    {
        SubnetId: "6201dcf2-1374-43ec-ae8b-78b4081572d3",
    },
}
isAutoRenewCreateInstancePeriodRequest:= int32(0)
promotionInfoCreateInstancePeriodRequest:= ""
commentCreateInstancePeriodRequest:= ""
request.Body = &model.CreateInstancePeriodRequest{
    IsAutoRenew: &isAutoRenewCreateInstancePeriodRequest,
    PromotionInfo: &promotionInfoCreateInstancePeriodRequest,
    Tags: &listTagsbody,
    ProductInfos: listProductInfosbody,
    SubscriptionNum: int32(1),
    PeriodNum: int32(1),
    PeriodType: int32(2),
    ChargingMode: int32(0),
    CloudServiceType: "hws.service.type.dbss",
    Region: "cn-north-7",
    Comment: &commentCreateInstancePeriodRequest,
    SecurityGroups: listSecurityGroupsbody,
    Nics: listNicsbody,
    EnterpriseProjectId: "0",
    AvailabilityZone: "cn-north-7c",
    VpcId: "4c035747-f77b-4c6d-b23b-cb3a2b96c7e6",
    Name: "DBSS-acc3",
    FlavorRef: "st6.xlarge.4",
}
response, err := client.CreateInstancesPeriodOrder(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	成功
400	失败
403	认证失败
500	服务端错误

错误码

请参见[错误码](#)。

3.3 数据库接口

3.3.1 查询数据库列表

功能介绍

查询数据库列表

调用方法

请参见[如何调用API](#)。

URI

GET /v1/{project_id}/{instance_id}/dbss/audit/databases

表 3-103 路径参数

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

表 3-104 Query 参数

参数	是否必选	参数类型	描述
status	否	String	实例状态 ON：开启 OFF：关闭
offset	否	String	偏移量
limit	否	String	查询记录数

请求参数

表 3-105 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

响应参数

状态码： 200

表 3-106 响应 Body 参数

参数	参数类型	描述
databases	Array of DataBaseBean objects	数据库信息列表
total	Integer	总数

表 3-107 DataBaseBean

参数	参数类型	描述
database	DataBase object	数据库信息

表 3-108 DataBase

参数	参数类型	描述
id	String	数据库ID
name	String	数据库名称
type	String	添加的数据库类型：枚举值：[] MYSQL [] ORACLE [] POSTGRESQL [] SQLSERVER [] DAMENG [] TAURUS [] DWS [] KINGBASE [] GAUSSDBOPENGAUSS [] GREENPLUM [] HIGHGO [] SHENTONG [] GBASE8A [] GBASE8S [] GBASEXDM [] MONGODB [] DDS
version	String	数据库版本
charset	String	数据库字符集
ip	String	数据库IP
port	String	数据库端口
os	String	数据库操作系统
status	String	开启状态（1：开启，0：关闭）
instance_name	String	数据库实例名
audit_status	String	数据库的运行状态 枚举值：[] ACTIVE [] SHUTOFF [] ERROR
agent_url	Array of strings	agent的唯一ID

参数	参数类型	描述
db_classification	String	数据库分类，取值范围：RDS（表示RDS数据库）和 ECS（自建数据库）

状态码：400

表 3-109 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-110 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码：403

表 3-111 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-112 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码：500

表 3-113 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-114 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

请求示例

```
/v1/{project_id}/{instance_id}/dbss/audit/databases
```

响应示例

状态码： 200

成功

```
{
  "databases": [ {
    "database": {
      "id": "zLkV83gBCwCqSg3BJt0m",
      "name": "db01",
      "type": "MYSQL",
      "version": "5.0",
      "charset": "UTF8",
      "ip": "192.168.0.204",
      "port": "3306",
      "os": "LINUX64",
      "status": "OFF",
      "instance_name": "",
      "audit_status": null,
      "agent_url": [ "zrKw83gBCwCqSg3Bkt1P" ],
      "db_classification": "ECS"
    }
  }
}]
```

状态码： 400

请求参数错误

```
{
  "error": {
    "error_code": "DBSS.XXXX",
    "error_msg": "XXX"
  }
}
```

状态码： 500

服务器内部错误

```
{
  "error": {
    "error_code": "DBSS.XXXX",
    "error_msg": "XXX"
  }
}
```

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.dbss.v1.region.DbssRegion;
import com.huaweicloud.sdk.dbss.v1.*;
import com.huaweicloud.sdk.dbss.v1.model.*;

public class ListAuditDatabasesSolution {

    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");

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

        DbssClient client = DbssClient.newBuilder()
            .withCredential(auth)
            .withRegion(DbssRegion.valueOf("cn-north-4"))
            .build();
        ListAuditDatabasesRequest request = new ListAuditDatabasesRequest();
        request.withStatus("<status>");
        request.withOffset("<offset>");
        request.withLimit("<limit>");
        try {
            ListAuditDatabasesResponse response = client.listAuditDatabases(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

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdbss.v1.region.dbss_region import DbssRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdbss.v1 import *

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

```
ak = os.getenv("CLOUD_SDK_AK")
sk = os.getenv("CLOUD_SDK_SK")

credentials = BasicCredentials(ak, sk) \

client = DbssClient.new_builder() \
    .with_credentials(credentials) \
    .with_region(DbssRegion.value_of("cn-north-4")) \
    .build()

try:
    request = ListAuditDatabasesRequest()
    request.status = "<status>"
    request.offset = "<offset>"
    request.limit = "<limit>"
    response = client.list_audit_databases(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"
    dbss "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/region"
)

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

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

    client := dbss.NewDbssClient(
        dbss.DbssClientBuilder().
            WithRegion(region.ValueOf("cn-north-4")).
            WithCredential(auth).
            Build())

    request := &model.ListAuditDatabasesRequest{}
    statusRequest := "<status>"
    request.Status = &statusRequest
    offsetRequest := "<offset>"
    request.Offset = &offsetRequest
    limitRequest := "<limit>"
    request.Limit = &limitRequest
    response, err := client.ListAuditDatabases(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	成功
400	请求参数错误
403	认证失败
500	服务器内部错误

错误码

请参见[错误码](#)。

3.3.2 添加 RDS 免 agent 数据库

功能介绍

添加RDS免agent数据库

调用方法

请参见[如何调用API](#)。

URI

POST /v1/{project_id}/{instance_id}/dbss/audit/databases/rds

表 3-115 路径参数

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

请求参数

表 3-116 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

表 3-117 请求 Body 参数

参数	是否必选	参数类型	描述
databases	是	Array of databases objects	添加数据库信息列表
total_count	否	Integer	总数

表 3-118 databases

参数	是否必选	参数类型	描述
id	是	String	数据库ID
db_name	是	String	数据库名称
status	是	String	数据库状态
port	是	String	数据库端口
ip	是	String	数据库IP
instance_name	是	String	数据库实例名称
version	是	String	数据库版本
type	是	String	数据库类型
enterprise_id	是	String	企业项目ID
enterprise_name	否	String	企业项目名称

响应参数

状态码： 200

表 3-119 响应 Body 参数

参数	参数类型	描述
illegal_db_id	Array of strings	添加失败的数据库实例id
legal_db_id	Array of strings	添加成功的数据库实例id

状态码： 400

表 3-120 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-121 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 403

表 3-122 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-123 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 500

表 3-124 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-125 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息。

请求示例

```
/v1/{project_id}/{instance_id}/dbss/audit/databases/rds
{
  "databases": [ {
    "id": "123751d3ee2f47aea64822e98318c6a8in01",
    "db_name": "rds1",
    "status": "ACTIVE",
    "port": "3306",
    "ip": "192.168.0.119",
    "instance_name": "rds1",
    "version": "8.0",
    "type": "MySQL",
    "enterprise_id": "0",
    "enterprise_name": "default"
  }, {
    "id": "2343f7285d684fed8b09fac201c3fc7ain01",
    "db_name": "rds2",
    "status": "ACTIVE",
    "port": "3306",
    "ip": "192.168.0.92",
    "instance_name": "rds2",
    "version": "8.0",
    "type": "MySQL",
    "enterprise_id": "0",
    "enterprise_name": "default"
  } ]
}
```

响应示例

状态码： 200

成功

```
{
  "illegal_db_id": [ ],
  "legal_db_id": [ "123751d3ee2f47aea64822e98318c6a8in01", "2343f7285d684fed8b09fac201c3fc7ain01" ]
}
```

状态码： 400

失败

```
{
  "error": {
    "error_code": "DBSS.XXXX",
    "error_msg": "XXX"
  }
}
```

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.dbss.v1.region.DbssRegion;
import com.huaweicloud.sdk.dbss.v1.*;
import com.huaweicloud.sdk.dbss.v1.model.*;

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

public class AddRdsNoAgentDatabaseSolution {

    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");

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

        DbssClient client = DbssClient.newBuilder()
            .withCredential(auth)
            .withRegion(DbssRegion.valueOf("cn-north-4"))
            .build();

        AddRdsNoAgentDatabaseRequest request = new AddRdsNoAgentDatabaseRequest();
        RdsNoAgentDbRequest body = new RdsNoAgentDbRequest();
        List<RdsNoAgentDbRequestDatabases> listbodyDatabases = new ArrayList<>();
        listbodyDatabases.add(
            new RdsNoAgentDbRequestDatabases()
                .withId("123751d3ee2f47aea64822e98318c6a8in01")
                .withDbName("rds1")
                .withStatus("ACTIVE")
                .withPort("3306")
                .withIp("192.168.0.119")
                .withInstanceName("rds1")
                .withVersion("8.0")
                .withType("MySQL")
                .withEnterpriseId("0")
                .withEnterpriseName("default")
        );
        listbodyDatabases.add(
            new RdsNoAgentDbRequestDatabases()
                .withId("2343f7285d684fed8b09fac201c3fc7ain01")
                .withDbName("rds2")
                .withStatus("ACTIVE")
                .withPort("3306")
                .withIp("192.168.0.92")
                .withInstanceName("rds2")
                .withVersion("8.0")
                .withType("MySQL")
                .withEnterpriseId("0")
                .withEnterpriseName("default")
        );
        body.withDatabases(listbodyDatabases);
        request.withBody(body);
        try {
            AddRdsNoAgentDatabaseResponse response = client.addRdsNoAgentDatabase(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  
  
from huaweicloudsdkcore.auth.credentials import BasicCredentials  
from huaweicloudsdkdbss.v1.region.dbss_region import DbssRegion  
from huaweicloudsdkcore.exceptions import exceptions  
from huaweicloudsdkdbss.v1 import *  
  
if __name__ == "__main__":  
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security  
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment  
    # variables and decrypted during use to ensure security.  
    # In this example, AK and SK are stored in environment variables for authentication. Before running this  
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment  
    ak = os.getenv("CLOUD_SDK_AK")  
    sk = os.getenv("CLOUD_SDK_SK")  
  
    credentials = BasicCredentials(ak, sk) \  
  
    client = DbssClient.new_builder() \  
        .with_credentials(credentials) \  
        .with_region(DbssRegion.value_of("cn-north-4")) \  
        .build()  
  
    try:  
        request = AddRdsNoAgentDatabaseRequest()  
        listDatabasesbody = [  
            RdsNoAgentDbRequestDatabases(  
                id="123751d3ee2f47aea64822e98318c6a8in01",  
                db_name="rds1",  
                status="ACTIVE",  
                port="3306",  
                ip="192.168.0.119",  
                instance_name="rds1",  
                version="8.0",  
                type="MySQL",  
                enterprise_id="0",  
                enterprise_name="default"  
            ),  
            RdsNoAgentDbRequestDatabases(  
                id="2343f7285d684fed8b09fac201c3fc7ain01",  
                db_name="rds2",  
                status="ACTIVE",  
                port="3306",  
                ip="192.168.0.92",  
                instance_name="rds2",  
                version="8.0",  
                type="MySQL",  
                enterprise_id="0",  
                enterprise_name="default"  
            )  
        ]  
        request.body = RdsNoAgentDbRequest(  
            databases=listDatabasesbody  
        )  
        response = client.add_rds_no_agent_database(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"
    dbss "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/region"
)

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

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

    client := dbss.NewDbssClient(
        dbss.DbssClientBuilder().
            WithRegion(region.ValueOf("cn-north-4")).
            WithCredential(auth).
            Build())

    request := &model.AddRdsNoAgentDatabaseRequest{
        enterpriseNameDatabases: "default"
        enterpriseNameDatabases1: "default"
        var listDatabasesbody = []model.RdsNoAgentDbRequestDatabases{
            {
                Id: "123751d3ee2f47aea64822e98318c6a8in01",
                DbName: "rds1",
                Status: "ACTIVE",
                Port: "3306",
                Ip: "192.168.0.119",
                InstanceName: "rds1",
                Version: "8.0",
                Type: "MySQL",
                EnterpriseId: "0",
                EnterpriseName: &enterpriseNameDatabases,
            },
            {
                Id: "2343f7285d684fed8b09fac201c3fc7ain01",
                DbName: "rds2",
                Status: "ACTIVE",
                Port: "3306",
                Ip: "192.168.0.92",
                InstanceName: "rds2",
                Version: "8.0",
                Type: "MySQL",
                EnterpriseId: "0",
                EnterpriseName: &enterpriseNameDatabases1,
            },
        },
    }
    request.Body = &model.RdsNoAgentDbRequest{
        Databases: listDatabasesbody,
    }
}
```

```
response, err := client.AddRdsNoAgentDatabase(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	成功
400	失败
403	认证失败
500	服务端错误

错误码

请参见[错误码](#)。

3.4 标签相关接口

3.4.1 查询项目标签

功能介绍

查询项目标签

调用方法

请参见[如何调用API](#)。

URI

GET /v1/{project_id}/{resource_type}/tags

表 3-126 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
resource_type	是	String	资源类型。审计：auditInstance

请求参数

表 3-127 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

响应参数

状态码： 200

表 3-128 响应 Body 参数

参数	参数类型	描述
tags	Array of tags objects	标签列表

表 3-129 tags

参数	参数类型	描述
key	String	键。最大长度128个字符。key满足3.1 KEY字符集规范。
values	Array of strings	值列表。每个值最大长度255个字符。value满足3.2 VALUE字符集规范。

状态码： 400

表 3-130 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-131 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 403

表 3-132 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-133 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 500

表 3-134 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-135 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

请求示例

```
/v1/{project_id}/{resource_type}/tags
```

响应示例

状态码： 200

成功

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

状态码: 400

失败

```
{
  "error": {
    "error_code": "DBSS.XXXX",
    "error_msg": "XXX"
  }
}
```

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.dbss.v1.region.DbssRegion;
import com.huaweicloud.sdk.dbss.v1.*;
import com.huaweicloud.sdk.dbss.v1.model.*;

public class ListProjectResourceTagsSolution {

    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");

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

        DbssClient client = DbssClient.newBuilder()
            .withCredential(auth)
            .withRegion(DbssRegion.valueOf("cn-north-4"))
            .build();
        ListProjectResourceTagsRequest request = new ListProjectResourceTagsRequest();
        try {
            ListProjectResourceTagsResponse response = client.listProjectResourceTags(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

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdbss.v1.region.dbss_region import DbssRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdbss.v1 import *

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

    credentials = BasicCredentials(ak, sk) \

    client = DbssClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DbssRegion.value_of("cn-north-4")) \
        .build()

    try:
        request = ListProjectResourceTagsRequest()
        response = client.list_project_resource_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

```
package main

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

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

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

```
Build()

client := dbss.NewDbssClient(
    dbss.DbssClientBuilder().
        WithRegion(region.ValueOf("cn-north-4")).
        WithCredential(auth).
        Build())

request := &model.ListProjectResourceTagsRequest{}
response, err := client.ListProjectResourceTags(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	成功
400	失败
403	认证失败
500	服务端错误

错误码

请参见[错误码](#)。

3.4.2 根据标签查询资源实例列表

功能介绍

根据标签查询资源实例列表

调用方法

请参见[如何调用API](#)。

URI

POST /v1/{project_id}/{resource_type}/resource-instances/filter

表 3-136 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
resource_type	是	String	资源类型。审计： auditInstance

表 3-137 Query 参数

参数	是否必选	参数类型	描述
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，偏移量（action为count时无此参数）从第一条数据偏移offset条数据后开始查询，如果action为filter默认为0（偏移0条数据，表示从第一条数据开始查询），必须为数字，不能为负数

请求参数

表 3-138 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

表 3-139 请求 Body 参数

参数	是否必选	参数类型	描述
matches	否	Array of matches objects	搜索字段,key为要匹配的字段,如resource_name等。value为匹配的值。key为固定字典值,不能包含重复的key或不支持的key。根据key的值确认是否需要模糊匹配,如resource_name默认为模糊搜索(不区分大小写),如果value为空字符串精确匹配(多数服务不存在资源名称为空的情况,因此此类情况返回空列表)。resource_id为精确匹配。第一期只做resource_name,后续再扩展。
not_tags	否	Array of TagKeyValue sBean objects	不包含标签,最多包含50个key,每个key下面的value最多10个,每个key对应的value可以为空数组但结构体不能缺失。Key不能重复,同一个key中values不能重复。结果返回不包含标签的资源列表,key之间是与的关系,key-value结构中value是或的关系。无过滤条件时返回全量数据
tags	否	Array of TagKeyValue sBean objects	包含标签,最多包含50个key,每个key下面的value最多10个,每个key对应的value可以为空数组但结构体不能缺失。Key不能重复,同一个key中values不能重复。结果返回包含所有标签的资源列表,key之间是与的关系,key-value结构中value是或的关系。无tag过滤条件时返回全量数据
tags_any	否	Array of TagKeyValue sBean objects	包含任意标签,最多包含50个key,每个key下面的value最多10个,每个key对应的value可以为空数组但结构体不能缺失。Key不能重复,同一个key中values不能重复。结果返回包含标签的资源列表,key之间是或的关系,key-value结构中value是或的关系。无过滤条件时返回全量数据

参数	是否必选	参数类型	描述
not_tags_any	否	Array of TagKeyValue sBean objects	不包含任意标签，最多包含50个key，每个key下面的value最多10个，每个key对应的value可以为空数组但结构体不能缺失。Key不能重复，同一个key中values不能重复。结果返回不包含标签的资源列表，key之间是或的关系，key-value结构中value是或的关系。无过滤条件时返回全量数据
sys_tags	否	TagKeyValue sBean object	仅op_service权限可以使用此字段做资源实例过滤条件。目前TMS调用时只包含一个tag结构体。key： _sys_enterprise_project_idvalue：企业项目id列表目前TMS调用时，key下面只包含一个value。0表示默认企业项目sys_tags和租户标签过滤条件 (without_any_tag、tags、tags_any、not_tags、not_tags_any)不能同时使用无sys_tags时按照tag接口处理，无tag过滤条件时返回全量数据
without_any_tag	否	Boolean	不包含任意一个标签，该字段为true时查询所有不带标签的资源，此时忽略“tags”、“tags_any”、“not_tags”、“not_tags_any”字段

表 3-140 matches

参数	是否必选	参数类型	描述
key	否	String	键
value	否	String	值

表 3-141 TagKeyValuesBean

参数	是否必选	参数类型	描述
key	是	String	键。最大长度128个unicode字符。key不能为空。(搜索时不 对此参数做字符集校验), key不 能为空或者空字符串, 不能为空 格, 校验和使用之前先trim 前 后半角空格
values	是	Array of strings	值列表。每个值最大长度255个 unicode字符, 校验和使用之前 先trim 前后半角空格。value可 为空数组但不可缺省。如果 values为空列表, 则表示 any_value (查询任意value)。 value之间为或的关系

响应参数

状态码: 200

表 3-142 响应 Body 参数

参数	参数类型	描述
resources	Array of resources objects	资源实例列表
total_count	Integer	总记录数

表 3-143 resources

参数	参数类型	描述
resource_detail	Object	资源详情。资源对象, 用于扩展。默认为空
resource_id	String	资源ID
resource_name	String	资源名称, 资源没有名称时默认为空字符串, eip 返回ip地址
tags	Array of tags objects	标签列表, 没有标签默认为空数组

参数	参数类型	描述
sys_tags	Array of sys_tags objects	仅op_service权限才可以获取此字段：目前只包含一个resource_tag 结构体 key: _sys_enterprise_project_id value: 企业项目id, 0表示默认企业项目 非op_service场景不能返回此字段

表 3-144 tags

参数	参数类型	描述
key	String	键
value	String	值

表 3-145 sys_tags

参数	参数类型	描述
key	String	键
value	String	值

状态码： 400

表 3-146 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-147 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 403

表 3-148 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-149 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 500

表 3-150 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-151 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

请求示例

```
/v1/{project_id}/{resource_type}/resource-instances/filter
```

```
{
  "matches": [ {
    "key": "resource_name",
    "value": "resource1"
  } ],
  "not_tags": [ {
    "key": "key1",
    "values": [ "*"value1", "value2" ]
  } ],
  "tags": [ {
    "key": "key1",
    "values": [ "*"value1", "value2" ]
  } ],
  "tags_any": [ {
    "key": "key1",
    "values": [ "value1", "value2" ]
  } ],
  "not_tags_any": [ {
    "key": "key1",
```

```
"values" : [ "value1", "value2" ]  
  },  
  "sys_tags" : [ {  
    "key" : "_sys_enterprise_project_id",  
    "values" : [ "5aa119a8-d25b-45a7-8d1b-88e127885635" ]  
  } ]  
}
```

响应示例

状态码： 200

成功

```
{  
  "resources" : [ {  
    "resource_detail" : null,  
    "resource_id" : "cdfs_cefs_wesas_12_dsad",  
    "resource_name" : "resouece1",  
    "tags" : [ {  
      "key" : "key1",  
      "value" : "value1"  
    }, {  
      "key" : "key2",  
      "value" : "value1"  
    } ],  
    "sys_tags" : [ {  
      "key" : "_sys_enterprise_project_id",  
      "value" : "5aa119a8-d25b-45a7-8d1b-88e127885635"  
    } ]  
  } ],  
  "total_count" : 1000  
}
```

状态码： 400

失败

```
{  
  "error" : {  
    "error_code" : "DBSS.XXXX",  
    "error_msg" : "XXX"  
  }  
}
```

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.dbss.v1.region.DbssRegion;  
import com.huaweicloud.sdk.dbss.v1.*;  
import com.huaweicloud.sdk.dbss.v1.model.*;  
  
import java.util.List;  
import java.util.ArrayList;  
  
public class ListResourceInstanceByTagSolution {
```

```
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");

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

    DbssClient client = DbssClient.newBuilder()
        .withCredential(auth)
        .withRegion(DbssRegion.valueOf("cn-north-4"))
        .build();

    ListResourceInstanceByTagRequest request = new ListResourceInstanceByTagRequest();
    request.withLimit("<limit>");
    request.withOffset("<offset>");
    ResourceInstanceTagRequest body = new ResourceInstanceTagRequest();
    List<String> listNotTagsAnyValues = new ArrayList<>();
    listNotTagsAnyValues.add("value1");
    listNotTagsAnyValues.add("value2");
    List<TagKeyValuesBean> listbodyNotTagsAny = new ArrayList<>();
    listbodyNotTagsAny.add(
        new TagKeyValuesBean()
            .withKey("key1")
            .withValues(listNotTagsAnyValues)
    );
    List<String> listTagsAnyValues = new ArrayList<>();
    listTagsAnyValues.add("value1");
    listTagsAnyValues.add("value2");
    List<TagKeyValuesBean> listbodyTagsAny = new ArrayList<>();
    listbodyTagsAny.add(
        new TagKeyValuesBean()
            .withKey("key1")
            .withValues(listTagsAnyValues)
    );
    List<String> listTagsValues = new ArrayList<>();
    listTagsValues.add("*value1");
    listTagsValues.add("value2");
    List<TagKeyValuesBean> listbodyTags = new ArrayList<>();
    listbodyTags.add(
        new TagKeyValuesBean()
            .withKey("key1")
            .withValues(listTagsValues)
    );
    List<String> listNotTagsValues = new ArrayList<>();
    listNotTagsValues.add("*value1");
    listNotTagsValues.add("value2");
    List<TagKeyValuesBean> listbodyNotTags = new ArrayList<>();
    listbodyNotTags.add(
        new TagKeyValuesBean()
            .withKey("key1")
            .withValues(listNotTagsValues)
    );
    List<ResourceInstanceTagRequestMatches> listbodyMatches = new ArrayList<>();
    listbodyMatches.add(
        new ResourceInstanceTagRequestMatches()
            .withKey("resource_name")
            .withValue("resource1")
    );
    body.withSysTags("[{\"values\": [\"5aa119a8-d25b-45a7-8d1b-88e127885635\"], \"key\": \"_sys_enterprise_project_id\"}]);");
    body.withNotTagsAny(listbodyNotTagsAny);
    body.withTagsAny(listbodyTagsAny);
    body.withTags(listbodyTags);
    body.withNotTags(listbodyNotTags);
}
```

```
body.withMatches(listbodyMatches);
request.withBody(body);
try {
    ListResourceInstanceByTagResponse response = client.listResourceInstanceByTag(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.getStatusCode());
    System.out.println(e.getRequestId());
    System.out.println(e.getErrorCode());
    System.out.println(e.getErrorMsg());
}
}
```

Python

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdbss.v1.region.dbss_region import DbssRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdbss.v1 import *

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

    credentials = BasicCredentials(ak, sk) \

    client = DbssClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DbssRegion.value_of("cn-north-4")) \
        .build()

    try:
        request = ListResourceInstanceByTagRequest()
        request.limit = "<limit>"
        request.offset = "<offset>"
        listValuesNotTagsAny = [
            "value1",
            "value2"
        ]
        listNotTagsAnybody = [
            TagKeyValuesBean(
                key="key1",
                values=listValuesNotTagsAny
            )
        ]
        listValuesTagsAny = [
            "value1",
            "value2"
        ]
        listTagsAnybody = [
            TagKeyValuesBean(
                key="key1",
                values=listValuesTagsAny
            )
        ]
        listValuesTags = [
```

```
        "value1",
        "value2"
    ]
    listTagsbody = [
        TagKeyValuesBean(
            key="key1",
            values=listValuesTags
        )
    ]
    listValuesNotTags = [
        "value1",
        "value2"
    ]
    listNotTagsbody = [
        TagKeyValuesBean(
            key="key1",
            values=listValuesNotTags
        )
    ]
    listMatchesbody = [
        ResourceInstanceTagRequestMatches(
            key="resource_name",
            value="resource1"
        )
    ]
    request.body = ResourceInstanceTagRequest(
        sys_tags="{\"values\": [\"5aa119a8-d25b-45a7-8d1b-88e127885635\"]}, {\"key\": \"_sys_enterprise_project_id\"}",
        not_tags_any=listNotTagsAnybody,
        tags_any=listTagsAnybody,
        tags=listTagsbody,
        not_tags=listNotTagsbody,
        matches=listMatchesbody
    )
    response = client.list_resource_instance_by_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

```
package main

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

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

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

    client := dbss.NewDbssClient(
```

```
dbss.DbssClientBuilder().
    WithRegion(region.ValueOf("cn-north-4")).
    WithCredential(auth).
    Build())

request := &model.ListResourceInstanceByTagRequest{
    limitRequest:= "<limit>"
    request.Limit = &limitRequest
    offsetRequest:= "<offset>"
    request.Offset = &offsetRequest
    var listValuesNotTagsAny = []string{
        "value1",
        "value2",
    }
    var listNotTagsAnybody = []model.TagKeyValuesBean{
        {
            Key: "key1",
            Values: listValuesNotTagsAny,
        },
    }
    var listValuesTagsAny = []string{
        "value1",
        "value2",
    }
    var listTagsAnybody = []model.TagKeyValuesBean{
        {
            Key: "key1",
            Values: listValuesTagsAny,
        },
    }
    var listValuesTags = []string{
        "value1",
        "value2",
    }
    var listTagsbody = []model.TagKeyValuesBean{
        {
            Key: "key1",
            Values: listValuesTags,
        },
    }
    var listValuesNotTags = []string{
        "value1",
        "value2",
    }
    var listNotTagsbody = []model.TagKeyValuesBean{
        {
            Key: "key1",
            Values: listValuesNotTags,
        },
    }
    keyMatches:= "resource_name"
    valueMatches:= "resource1"
    var listMatchesbody = []model.ResourceInstanceTagRequestMatches{
        {
            Key: &keyMatches,
            Value: &valueMatches,
        },
    }
    var sysTagsSysTags interface{} = "[{"values":["5aa119a8-d25b-45a7-8d1b-88e127885635"],"key
    \":\"_sys_enterprise_project_id\"}]]"
    request.Body = &model.ResourceInstanceTagRequest{
        SysTags: &sysTagsSysTags,
        NotTagsAny: &listNotTagsAnybody,
        TagsAny: &listTagsAnybody,
        Tags: &listTagsbody,
        NotTags: &listNotTagsbody,
        Matches: &listMatchesbody,
    }
    response, err := client.ListResourceInstanceByTag(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	成功
400	失败
403	认证失败
500	服务端错误

错误码

请参见[错误码](#)。

3.4.3 根据标签查询资源实例数量

功能介绍

根据标签查询资源实例数量

调用方法

请参见[如何调用API](#)。

URI

POST /v1/{project_id}/{resource_type}/resource-instances/count

表 3-152 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
resource_type	是	String	资源类型。审计： auditInstance

请求参数

表 3-153 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

表 3-154 请求 Body 参数

参数	是否必选	参数类型	描述
matches	否	Array of matches objects	搜索字段,key为要匹配的字段,如resource_name等。value为匹配的值。key为固定字典值,不能包含重复的key或不支持的key。根据key的值确认是否需要模糊匹配,如resource_name默认为模糊搜索(不区分大小写),如果value为空字符串精确匹配(多数服务不存在资源名称为空的情况,因此此类情况返回空列表)。resource_id为精确匹配。第一期只做resource_name,后续再扩展。
not_tags	否	Array of TagKeyValue sBean objects	不包含标签,最多包含50个key,每个key下面的value最多10个,每个key对应的value可以为空数组但结构体不能缺失。Key不能重复,同一个key中values不能重复。结果返回不包含标签的资源列表,key之间是与的关系,key-value结构中value是或的关系。无过滤条件时返回全量数据
tags	否	Array of TagKeyValue sBean objects	包含标签,最多包含50个key,每个key下面的value最多10个,每个key对应的value可以为空数组但结构体不能缺失。Key不能重复,同一个key中values不能重复。结果返回包含所有标签的资源列表,key之间是与的关系,key-value结构中value是或的关系。无tag过滤条件时返回全量数据

参数	是否必选	参数类型	描述
tags_any	否	Array of TagKeyValue sBean objects	包含任意标签，最多包含50个key，每个key下面的value最多10个，每个key对应的value可以为空数组但结构体不能缺失。Key不能重复，同一个key中values不能重复。结果返回包含标签的资源列表，key之间是或的关系，key-value结构中value是或的关系。无过滤条件时返回全量数据
not_tags_any	否	Array of TagKeyValue sBean objects	不包含任意标签，最多包含50个key，每个key下面的value最多10个，每个key对应的value可以为空数组但结构体不能缺失。Key不能重复，同一个key中values不能重复。结果返回不包含标签的资源列表，key之间是或的关系，key-value结构中value是或的关系。无过滤条件时返回全量数据
sys_tags	否	TagKeyValue sBean object	仅op_service权限可以使用此字段做资源实例过滤条件。目前TMS调用时只包含一个tag结构体。key: _sys_enterprise_project_idvalue：企业项目id列表目前TMS调用时，key下面只包含一个value。0表示默认企业项目sys_tags和租户标签过滤条件 (without_any_tag、tags、tags_any、not_tags、not_tags_any)不能同时使用无sys_tags时按照tag接口处理，无tag过滤条件时返回全量数据
without_any_tag	否	Boolean	不包含任意一个标签，该字段为true时查询所有不带标签的资源，此时忽略“tags”、“tags_any”、“not_tags”、“not_tags_any”字段

表 3-155 matches

参数	是否必选	参数类型	描述
key	否	String	键
value	否	String	值

表 3-156 TagKeyValuesBean

参数	是否必选	参数类型	描述
key	是	String	键。最大长度128个unicode字符。key不能为空。(搜索时不 对此参数做字符集校验), key不 能为空或者空字符串, 不能为空 格, 校验和使用之前先trim 前 后半角空格
values	是	Array of strings	值列表。每个值最大长度255个 unicode字符, 校验和使用之前 先trim 前后半角空格。value可 为空数组但不可缺省。如果 values为空列表, 则表示 any_value (查询任意value)。 value之间为或的关系

响应参数

状态码: 200

表 3-157 响应 Body 参数

参数	参数类型	描述
resources	Array of resources objects	资源实例列表
total_count	Integer	总记录数

表 3-158 resources

参数	参数类型	描述
resource_detail	Object	资源详情。资源对象, 用于扩展。默认为空
resource_id	String	资源ID
resource_name	String	资源名称, 资源没有名称时默认为空字符串, eip 返回ip地址
tags	Array of tags objects	标签列表, 没有标签默认为空数组

参数	参数类型	描述
sys_tags	Array of sys_tags objects	仅op_service权限才可以获取此字段： 目前只包含一个resource_tag 结构体 key: _sys_enterprise_project_id value: 企业项目id, 0表示默认企业项目 非op_service场景不能返回此字段

表 3-159 tags

参数	参数类型	描述
key	String	键
value	String	值

表 3-160 sys_tags

参数	参数类型	描述
key	String	键
value	String	值

状态码： 400

表 3-161 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-162 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 403

表 3-163 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-164 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 500

表 3-165 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-166 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

请求示例

```
/v1/{project_id}/{resource_type}/resource-instances/count
```

```
{
  "matches": [ {
    "key": "resource_name",
    "value": "resource1"
  } ],
  "not_tags": [ {
    "key": "key1",
    "values": [ "*"value1", "value2" ]
  } ],
  "tags": [ {
    "key": "key1",
    "values": [ "*"value1", "value2" ]
  } ],
  "tags_any": [ {
    "key": "key1",
    "values": [ "value1", "value2" ]
  } ],
  "not_tags_any": [ {
    "key": "key1",
```

```
"values" : [ "value1", "value2" ]
  },
  "sys_tags" : [ {
    "key" : "_sys_enterprise_project_id",
    "values" : [ "5aa119a8-d25b-45a7-8d1b-88e127885635" ]
  } ]
}
```

响应示例

状态码： 200

成功

```
{
  "total_count" : 1000
}
```

状态码： 400

失败

```
{
  "error" : {
    "error_code" : "DBSS.XXXX",
    "error_msg" : "XXX"
  }
}
```

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.dbss.v1.region.DbssRegion;
import com.huaweicloud.sdk.dbss.v1.*;
import com.huaweicloud.sdk.dbss.v1.model.*;

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

public class CountResourceInstanceByTagSolution {

    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");

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

        DbssClient client = DbssClient.newBuilder()
            .withCredential(auth)
            .withRegion(DbssRegion.valueOf("cn-north-4"))
```

```
        .build();
        CountResourceInstanceByTagRequest request = new CountResourceInstanceByTagRequest();
        ResourceInstanceTagRequest body = new ResourceInstanceTagRequest();
        List<String> listNotTagsAnyValues = new ArrayList<>();
        listNotTagsAnyValues.add("value1");
        listNotTagsAnyValues.add("value2");
        List<TagKeyValuesBean> listbodyNotTagsAny = new ArrayList<>();
        listbodyNotTagsAny.add(
            new TagKeyValuesBean()
                .withKey("key1")
                .withValues(listNotTagsAnyValues)
        );
        List<String> listTagsAnyValues = new ArrayList<>();
        listTagsAnyValues.add("value1");
        listTagsAnyValues.add("value2");
        List<TagKeyValuesBean> listbodyTagsAny = new ArrayList<>();
        listbodyTagsAny.add(
            new TagKeyValuesBean()
                .withKey("key1")
                .withValues(listTagsAnyValues)
        );
        List<String> listTagsValues = new ArrayList<>();
        listTagsValues.add("*value1");
        listTagsValues.add("value2");
        List<TagKeyValuesBean> listbodyTags = new ArrayList<>();
        listbodyTags.add(
            new TagKeyValuesBean()
                .withKey("key1")
                .withValues(listTagsValues)
        );
        List<String> listNotTagsValues = new ArrayList<>();
        listNotTagsValues.add("value1");
        listNotTagsValues.add("value2");
        List<TagKeyValuesBean> listbodyNotTags = new ArrayList<>();
        listbodyNotTags.add(
            new TagKeyValuesBean()
                .withKey("key1")
                .withValues(listNotTagsValues)
        );
        List<ResourceInstanceTagRequestMatches> listbodyMatches = new ArrayList<>();
        listbodyMatches.add(
            new ResourceInstanceTagRequestMatches()
                .withKey("resource_name")
                .withValue("resource1")
        );
        body.withSysTags("{\"values\": [\"5aa119a8-d25b-45a7-8d1b-88e127885635\"], \"key\": \"_sys_enterprise_project_id\"}");
        body.withNotTagsAny(listbodyNotTagsAny);
        body.withTagsAny(listbodyTagsAny);
        body.withTags(listbodyTags);
        body.withNotTags(listbodyNotTags);
        body.withMatches(listbodyMatches);
        request.withBody(body);
        try {
            CountResourceInstanceByTagResponse response = client.countResourceInstanceByTag(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

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdbss.v1.region.dbss_region import DbssRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdbss.v1 import *

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

    credentials = BasicCredentials(ak, sk) \

    client = DbssClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DbssRegion.value_of("cn-north-4")) \
        .build()

    try:
        request = CountResourceInstanceByTagRequest()
        listValuesNotTagsAny = [
            "value1",
            "value2"
        ]
        listNotTagsAnybody = [
            TagKeyValuesBean(
                key="key1",
                values=listValuesNotTagsAny
            )
        ]
        listValuesTagsAny = [
            "value1",
            "value2"
        ]
        listTagsAnybody = [
            TagKeyValuesBean(
                key="key1",
                values=listValuesTagsAny
            )
        ]
        listValuesTags = [
            "value1",
            "value2"
        ]
        listTagsbody = [
            TagKeyValuesBean(
                key="key1",
                values=listValuesTags
            )
        ]
        listValuesNotTags = [
            "value1",
            "value2"
        ]
        listNotTagsbody = [
            TagKeyValuesBean(
                key="key1",
                values=listValuesNotTags
            )
        ]
        listMatchesbody = [
            ResourceInstanceTagRequestMatches(
                key="resource_name",
```

```
        value="resource1"
    )
]
request.body = ResourceInstanceTagRequest(
    sys_tags=["values":["5aa119a8-d25b-45a7-8d1b-88e127885635"],"key
\":"_sys_enterprise_project_id"}],
    not_tags_any=listNotTagsAnybody,
    tags_any=listTagsAnybody,
    tags=listTagsbody,
    not_tags=listNotTagsbody,
    matches=listMatchesbody
)
response = client.count_resource_instance_by_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

```
package main

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

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

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

    client := dbss.NewDbssClient(
        dbss.DbssClientBuilder().
            WithRegion(region.ValueOf("cn-north-4")).
            WithCredential(auth).
            Build())

    request := &model.CountResourceInstanceByTagRequest{}
    var listValuesNotTagsAny = []string{
        "value1",
        "value2",
    }
    var listNotTagsAnybody = []model.TagKeyValuesBean{
        {
            Key: "key1",
            Values: listValuesNotTagsAny,
        },
    }
    var listValuesTagsAny = []string{
        "value1",
        "value2",
    }
    var listTagsAnybody = []model.TagKeyValuesBean{
        {
```



```

        Key: "key1",
        Values: listValuesTagsAny,
    },
}
var listValuesTags = []string{
    "*value1",
    "value2",
}
var listTagsbody = []model.TagKeyValuesBean{
    {
        Key: "key1",
        Values: listValuesTags,
    },
}
var listValuesNotTags = []string{
    "*value1",
    "value2",
}
var listNotTagsbody = []model.TagKeyValuesBean{
    {
        Key: "key1",
        Values: listValuesNotTags,
    },
}
keyMatches:= "resource_name"
valueMatches:= "resource1"
var listMatchesbody = []model.ResourceInstanceTagRequestMatches{
    {
        Key: &keyMatches,
        Value: &valueMatches,
    },
}
}
var sysTagsSysTags interface{} = "[{\\"values\\":[\"5aa119a8-d25b-45a7-8d1b-88e127885635\\\"],\\"key
\\":\\"_sys_enterprise_project_id\\"}]"
request.Body = &model.ResourceInstanceTagRequest{
    SysTags: &sysTagsSysTags,
    NotTagsAny: &listNotTagsAnybody,
    TagsAny: &listTagsAnybody,
    Tags: &listTagsbody,
    NotTags: &listNotTagsbody,
    Matches: &listMatchesbody,
}
response, err := client.CountResourceInstanceByTag(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	成功
400	失败
403	认证失败

状态码	描述
500	服务端错误

错误码

请参见[错误码](#)。

3.4.4 批量添加资源标签

功能介绍

批量添加资源标签

调用方法

请参见[如何调用API](#)。

URI

POST /v1/{project_id}/{resource_type}/{resource_id}/tags/create

表 3-167 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
resource_type	是	String	资源类型。审计： auditInstance
resource_id	是	String	资源ID

请求参数

表 3-168 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

表 3-169 请求 Body 参数

参数	是否必选	参数类型	描述
tags	否	Array of KeyValueBean objects	标签列表 租户权限时该字段必选, op_service权限时和 sys_tags二选一
sys_tags	否	Array of KeyValueBean objects	系统标签列表 op_service权限可以访问, 和tags二选一。目前TMS调用时只包含一个 resource_tag结构体, key固定为: _sys_enterprise_project_id value是UUID或0,value为0表示默认企业项目

表 3-170 KeyValueBean

参数	是否必选	参数类型	描述
key	是	String	键
value	是	String	值

响应参数

状态码: 400

表 3-171 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-172 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码: 403

表 3-173 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-174 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 500

表 3-175 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-176 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

请求示例

```
/v1/{project_id}/{resource_type}/{resource_id}/tags/create
{
  "tags": [ {
    "key": "key1"
  }, {
    "key": "key2",
    "value": "value3"
  } ],
  "sys_tags": [ {
    "key": "_sys_enterprise_project_id",
    "value": "5aa119a8-d25b-45a7-8d1b-88e127885635"
  } ]
}
```

响应示例

状态码： 400

失败

```
{
  "error" : {
    "error_code" : "DBSS.XXXX",
    "error_msg" : "XXX"
  }
}
```

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.dbss.v1.region.DbssRegion;
import com.huaweicloud.sdk.dbss.v1.*;
import com.huaweicloud.sdk.dbss.v1.model.*;

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

public class BatchAddResourceTagSolution {

    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");

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

        DbssClient client = DbssClient.newBuilder()
            .withCredential(auth)
            .withRegion(DbssRegion.valueOf("cn-north-4"))
            .build();
        BatchAddResourceTagRequest request = new BatchAddResourceTagRequest();
        ResourceTagRequest body = new ResourceTagRequest();
        List<KeyValueBean> listbodySysTags = new ArrayList<>();
        listbodySysTags.add(
            new KeyValueBean()
                .withKey("_sys_enterprise_project_id")
                .withValue("5aa119a8-d25b-45a7-8d1b-88e127885635")
        );
        List<KeyValueBean> listbodyTags = new ArrayList<>();
        listbodyTags.add(
            new KeyValueBean()
                .withKey("key1")
        );
        listbodyTags.add(
            new KeyValueBean()
                .withKey("key2")
                .withValue("value3")
        );
        body.withSysTags(listbodySysTags);
        body.withTags(listbodyTags);
        request.withBody(body);
        try {
```

```
BatchAddResourceTagResponse response = client.batchAddResourceTag(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

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdbss.v1.region.dbss_region import DbssRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdbss.v1 import *

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

    credentials = BasicCredentials(ak, sk) \

    client = DbssClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DbssRegion.value_of("cn-north-4")) \
        .build()

    try:
        request = BatchAddResourceTagRequest()
        listSysTagsbody = [
            KeyValueBean(
                key="_sys_enterprise_project_id",
                value="5aa119a8-d25b-45a7-8d1b-88e127885635"
            )
        ]
        listTagsbody = [
            KeyValueBean(
                key="key1"
            ),
            KeyValueBean(
                key="key2",
                value="value3"
            )
        ]
        request.body = ResourceTagRequest(
            sys_tags=listSysTagsbody,
            tags=listTagsbody
        )
        response = client.batch_add_resource_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

```
package main

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

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

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

    client := dbss.NewDbssClient(
        dbss.DbssClientBuilder().
            WithRegion(region.ValueOf("cn-north-4")).
            WithCredential(auth).
            Build())

    request := &model.BatchAddResourceTagRequest{}
    var listSysTagsbody = []model.KeyValueBean{
        {
            Key: "_sys_enterprise_project_id",
            Value: "5aa119a8-d25b-45a7-8d1b-88e127885635",
        },
    }
    var listTagsbody = []model.KeyValueBean{
        {
            Key: "key1",
        },
        {
            Key: "key2",
            Value: "value3",
        },
    }
    request.Body = &model.ResourceTagRequest{
        SysTags: &listSysTagsbody,
        Tags: &listTagsbody,
    }
    response, err := client.BatchAddResourceTag(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	成功
400	失败
403	认证失败
500	服务端错误

错误码

请参见[错误码](#)。

3.4.5 批量删除资源标签

功能介绍

批量删除资源标签

调用方法

请参见[如何调用API](#)。

URI

DELETE /v1/{project_id}/{resource_type}/{resource_id}/tags/delete

表 3-177 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
resource_type	是	String	资源类型。审计： auditInstance
resource_id	是	String	资源ID

请求参数

表 3-178 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

表 3-179 请求 Body 参数

参数	是否必选	参数类型	描述
tags	否	Array of KeyValueBean objects	标签列表 租户权限时该字段必选, op_service权限时和 sys_tags二选一
sys_tags	否	Array of KeyValueBean objects	系统标签列表 op_service权限可以访问, 和tags二选一。目前TMS调用时只包含一个 resource_tag结构体, key固定为: _sys_enterprise_project_id value是UUID或0,value为0表示默认企业项目

表 3-180 KeyValueBean

参数	是否必选	参数类型	描述
key	是	String	键
value	是	String	值

响应参数

状态码: 400

表 3-181 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-182 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码: 403

表 3-183 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-184 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 500

表 3-185 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-186 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

请求示例

```
/v1/{project_id}/{resource_type}/{resource_id}/tags/delete
```

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

响应示例

状态码： 400

失败

```
{  
  "error": {  
    "error_code": "DBSS.XXXX",  
  }  
}
```

```
"error_msg" : "XXX"  
}  
}
```

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.dbss.v1.region.DbssRegion;  
import com.huaweicloud.sdk.dbss.v1.*;  
import com.huaweicloud.sdk.dbss.v1.model.*;  
  
import java.util.List;  
import java.util.ArrayList;  
  
public class BatchDeleteResourceTagSolution {  
    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");  
  
        ICredential auth = new BasicCredentials()  
            .withAk(ak)  
            .withSk(sk);  
  
        DbssClient client = DbssClient.newBuilder()  
            .withCredential(auth)  
            .withRegion(DbssRegion.valueOf("cn-north-4"))  
            .build();  
        BatchDeleteResourceTagRequest request = new BatchDeleteResourceTagRequest();  
        ResourceTagRequest body = new ResourceTagRequest();  
        List<KeyValueBean> listbodyTags = new ArrayList<>();  
        listbodyTags.add(  
            new KeyValueBean()  
                .withKey("key1")  
        );  
        listbodyTags.add(  
            new KeyValueBean()  
                .withKey("key2")  
                .withValue("value3")  
        );  
        body.withTags(listbodyTags);  
        request.withBody(body);  
        try {  
            BatchDeleteResourceTagResponse response = client.batchDeleteResourceTag(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

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdbss.v1.region.dbss_region import DbssRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdbss.v1 import *

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

    credentials = BasicCredentials(ak, sk) \

    client = DbssClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DbssRegion.value_of("cn-north-4")) \
        .build()

    try:
        request = BatchDeleteResourceTagRequest()
        listTagsbody = [
            KeyValueBean(
                key="key1"
            ),
            KeyValueBean(
                key="key2",
                value="value3"
            )
        ]
        request.body = ResourceTagRequest(
            tags=listTagsbody
        )
        response = client.batch_delete_resource_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

```
package main

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

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

```
// In this example, AK and SK are stored in environment variables for authentication. Before running this
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
ak := os.Getenv("CLOUD_SDK_AK")
sk := os.Getenv("CLOUD_SDK_SK")

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

client := dbss.NewDbssClient(
    dbss.DbssClientBuilder().
        WithRegion(region.ValueOf("cn-north-4")).
        WithCredential(auth).
        Build())

request := &model.BatchDeleteResourceTagRequest{
var listTagsbody = []model.KeyValueBean{
    {
        Key: "key1",
    },
    {
        Key: "key2",
        Value: "value3",
    },
}
request.Body = &model.ResourceTagRequest{
    Tags: &listTagsbody,
}
response, err := client.BatchDeleteResourceTag(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	成功
400	失败
403	认证失败
500	服务端错误

错误码

请参见[错误码](#)。

3.5 实例侧规则接口

3.5.1 查询审计范围策略列表

功能介绍

查询审计范围策略列表

调用方法

请参见[如何调用API](#)。

URI

GET /v1/{project_id}/{instance_id}/dbss/audit/rule/scopes

表 3-187 路径参数

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

表 3-188 Query 参数

参数	是否必选	参数类型	描述
offset	否	String	偏移量
limit	否	String	查询记录数

请求参数

表 3-189 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

响应参数

状态码： 200

表 3-190 响应 Body 参数

参数	参数类型	描述
scopes	Array of RuleScopeInfo objects	审计范围规则列表
total	Integer	总数

表 3-191 RuleScopeInfo

参数	参数类型	描述
id	String	审计范围规则ID
name	String	审计范围名称
action	String	审计范围动作
status	String	审计范围规则状态
exception_ips	String	审计范围例外IP
source_ips	String	审计范围规则源IP
source_ports	String	审计范围源端口
db_ids	String	数据库ID
db_names	String	数据库名称
db_users	String	数据库用户
all_audit	Boolean	是否全审计

状态码： 400

表 3-192 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-193 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 403

表 3-194 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-195 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 500

表 3-196 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-197 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

请求示例

```
/v1/{project_id}/{instance_id}/dbss/audit/rule/scopes
```

响应示例

状态码： 200

成功

```
{  
  "scopes": [ {  
    "id": "zX4W2ngBo47GiyUSBuNs",  
    "name": "全审计规则",  
    "action": "",  
    "status": "ON",  
    "exception_ips": "",  
    "source_ips": "",  
    "source_ports": "",
```



```
"db_ids" : "",
"db_names" : "",
"db_users" : "",
"all_audit" : true
}],
"total" : 1
}
```

状态码： 400

请求参数错误

```
{
  "error" : {
    "error_code" : "DBSS.XXXX",
    "error_msg" : "XXX"
  }
}
```

状态码： 500

服务器内部错误

```
{
  "error" : {
    "error_code" : "DBSS.XXXX",
    "error_msg" : "XXX"
  }
}
```

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.dbss.v1.region.DbssRegion;
import com.huaweicloud.sdk.dbss.v1.*;
import com.huaweicloud.sdk.dbss.v1.model.*;

public class ListAuditRuleScopesSolution {

    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");

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

        DbssClient client = DbssClient.newBuilder()
            .withCredential(auth)
            .withRegion(DbssRegion.valueOf("cn-north-4"))
            .build();
        ListAuditRuleScopesRequest request = new ListAuditRuleScopesRequest();
```

```
request.withOffset("<offset>");
request.withLimit("<limit>");
try {
    ListAuditRuleScopesResponse response = client.listAuditRuleScopes(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.getStatusCode());
    System.out.println(e.getRequestId());
    System.out.println(e.getErrorCode());
    System.out.println(e.getErrorMsg());
}
}
```

Python

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdbss.v1.region.dbss_region import DbssRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdbss.v1 import *

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

    credentials = BasicCredentials(ak, sk) \

    client = DbssClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DbssRegion.value_of("cn-north-4")) \
        .build()

    try:
        request = ListAuditRuleScopesRequest()
        request.offset = "<offset>"
        request.limit = "<limit>"
        response = client.list_audit_rule_scopes(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"
    dbss "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/region"
)

func main() {
```

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

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

client := dbss.NewDbssClient(
    dbss.DbssClientBuilder().
        WithRegion(region.ValueOf("cn-north-4")).
        WithCredential(auth).
        Build())

request := &model.ListAuditRuleScopesRequest{
    offsetRequest:= "<offset>"
    request.Offset = &offsetRequest
    limitRequest:= "<limit>"
    request.Limit = &limitRequest
    response, err := client.ListAuditRuleScopes(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	成功
400	请求参数错误
403	认证失败
500	服务器内部错误

错误码

请参见[错误码](#)。

3.5.2 查询 SQL 注入规则策略

功能介绍

查询SQL注入规则策略

调用方法

请参见[如何调用API](#)。

URI

POST /v1/{project_id}/{instance_id}/dbss/audit/rule/sql-injections

表 3-198 路径参数

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

请求参数

表 3-199 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

表 3-200 请求 Body 参数

参数	是否必选	参数类型	描述
risk_levels	否	String	风险级别:(多项查询使用逗号分隔) <input type="checkbox"/> HIGH <input type="checkbox"/> MEDIUM <input type="checkbox"/> LOW <input type="checkbox"/> NO_RISK

响应参数

状态码: 200

表 3-201 响应 Body 参数

参数	参数类型	描述
rules	Array of rules objects	SQL规则列表
total	Integer	当前范围总数

表 3-202 rules

参数	参数类型	描述
id	String	SQL规则ID
name	String	SQL规则名称
status	String	规则的状态： <input type="checkbox"/> ON <input type="checkbox"/> OFF
risk_level	String	风险级别 <input type="checkbox"/> HIGH <input type="checkbox"/> MEDIUM <input type="checkbox"/> LOW
type	String	类型
rank	Integer	等级
feature	String	SQL命令特征
regex	String	正则表达式

状态码： 400

表 3-203 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-204 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 403

表 3-205 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-206 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息。

状态码： 500

表 3-207 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-208 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

请求示例

```
/v1/{project_id}/{instance_id}/dbss/audit/rule/sql-injections  
  
{  
  "risk_levels": "HIGH"  
}
```

响应示例

状态码： 200

成功

```
{  
  "rules": [ {  
    "id": "zX4W2ngBo47GiyUSBuNs",  
    "name": "MYSQL报错型SQL注入",  
    "status": "ON",  
    "type": "SYSTEM",  
    "risk_level": "HIGH",  
    "rank": 1,  
    "feature": "正则表达式",  
    "regex": "(.*)?(select)\\s+[0-9]+\\s+from\\s+\\s+\\s+\\s+select\\s+count(.*)?(concat)\\s*(.*)?(from)\\s*(information_schema.tables)(.*)?(group)\\s+(by)(.*)?"  
  } ],  
  "total": 1  
}
```

状态码： 400

请求参数错误

```
{  
  "error": {
```

```
"error_code" : "DBSS.XXXX",  
"error_msg" : "XXX"  
}  
}
```

状态码： 500

服务器内部错误

```
{  
  "error" : {  
    "error_code" : "DBSS.XXXX",  
    "error_msg" : "XXX"  
  }  
}
```

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.dbss.v1.region.DbssRegion;  
import com.huaweicloud.sdk.dbss.v1.*;  
import com.huaweicloud.sdk.dbss.v1.model.*;  
  
public class ListSqlInjectionRulesSolution {  
  
    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");  
  
        ICredential auth = new BasicCredentials()  
            .withAk(ak)  
            .withSk(sk);  
  
        DbssClient client = DbssClient.newBuilder()  
            .withCredential(auth)  
            .withRegion(DbssRegion.valueOf("cn-north-4"))  
            .build();  
        ListSqlInjectionRulesRequest request = new ListSqlInjectionRulesRequest();  
        SqlRuleRequest body = new SqlRuleRequest();  
        body.withRiskLevels("HIGH");  
        request.withBody(body);  
        try {  
            ListSqlInjectionRulesResponse response = client.listSqlInjectionRules(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

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdbss.v1.region.dbss_region import DbssRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdbss.v1 import *

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

    credentials = BasicCredentials(ak, sk) \

    client = DbssClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DbssRegion.value_of("cn-north-4")) \
        .build()

    try:
        request = ListSqlInjectionRulesRequest()
        request.body = SqlRuleRequest(
            risk_levels="HIGH"
        )
        response = client.list_sql_injection_rules(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"
    dbss "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/region"
)

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

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



```
client := dbss.NewDbssClient(
    dbss.DbssClientBuilder().
        WithRegion(region.ValueOf("cn-north-4")).
        WithCredential(auth).
        Build())

request := &model.ListSqlInjectionRulesRequest{
    riskLevelsSqlRuleRequest:= "HIGH"
    request.Body = &model.SqlRuleRequest{
        RiskLevels: &riskLevelsSqlRuleRequest,
    }
}
response, err := client.ListSqlInjectionRules(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	成功
400	请求参数错误
403	认证失败
500	服务器内部错误

错误码

请参见[错误码](#)。

3.5.3 查询风险规则策略

功能介绍

查询风险规则策略

调用方法

请参见[如何调用API](#)。

URI

GET /v1/{project_id}/{instance_id}/dbss/audit/rule/risk

表 3-209 路径参数

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

表 3-210 Query 参数

参数	是否必选	参数类型	描述
name	否	String	风险名称
risk_levels	否	String	风险级别 [LOW,MEDIUM,HIGH,NO_RISK]

请求参数

表 3-211 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

响应参数

状态码： 200

表 3-212 响应 Body 参数

参数	参数类型	描述
rules	Array of rules objects	风险规则列表
total	Integer	总数

表 3-213 rules

参数	参数类型	描述
id	String	风险规则ID
name	String	风险规则名称
type	String	风险类型

参数	参数类型	描述
feature	String	风险特征
status	String	风险规则状态
rank	Integer	风险规则优先级
risk_level	String	风险级别

状态码： 400

表 3-214 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-215 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 403

表 3-216 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-217 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 500

表 3-218 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-219 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

请求示例

```
/v1/{project_id}/{instance_id}/dbss/audit/rule/risk
```

响应示例

状态码：200

成功

```
{
  "rules": [ {
    "id": "x4W2ngBo47GiyUSBeOy",
    "name": "Database_drag_detection",
    "type": "OPERATE",
    "feature": "CLIENT[Any]OPERATE[[SELECT]OBJECT[Any]",
    "status": "ON",
    "rank": -1,
    "risk_level": "HIGH"
  }, {
    "id": "xn4W2ngBo47GiyUSBeP4",
    "name": "Database_Slow_SQL_Detection",
    "type": "OPERATE",
    "feature": "CLIENT[Any]OPERATE[[SELECT]OBJECT[Any]",
    "status": "ON",
    "rank": -2,
    "risk_level": "LOW"
  } ],
  "total": 2
}
```

状态码：400

请求参数错误

```
{
  "error": {
    "error_code": "DBSS.XXXX",
    "error_msg": "XXX"
  }
}
```

状态码：500

服务器内部错误

```
{
  "error": {
```

```
"error_code" : "DBSS.XXXX",  
"error_msg" : "XXX"  
}  
}
```

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.dbss.v1.region.DbssRegion;  
import com.huaweicloud.sdk.dbss.v1.*;  
import com.huaweicloud.sdk.dbss.v1.model.*;  
  
public class ListAuditRuleRisksSolution {  
  
    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");  
  
        ICredential auth = new BasicCredentials()  
            .withAk(ak)  
            .withSk(sk);  
  
        DbssClient client = DbssClient.newBuilder()  
            .withCredential(auth)  
            .withRegion(DbssRegion.valueOf("cn-north-4"))  
            .build();  
        ListAuditRuleRisksRequest request = new ListAuditRuleRisksRequest();  
        request.withName("<name>");  
        request.withRiskLevels("<risk_levels>");  
        try {  
            ListAuditRuleRisksResponse response = client.listAuditRuleRisks(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.getStatusCode());  
            System.out.println(e.getRequestId());  
            System.out.println(e.getErrorCode());  
            System.out.println(e.getErrorMsg());  
        }  
    }  
}
```

Python

```
# coding: utf-8  
  
from huaweicloudsdkcore.auth.credentials import BasicCredentials
```

```
from huaweicloudsdkdbss.v1.region.dbss_region import DbssRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdbss.v1 import *

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

    credentials = BasicCredentials(ak, sk) \

    client = DbssClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DbssRegion.value_of("cn-north-4")) \
        .build()

    try:
        request = ListAuditRuleRisksRequest()
        request.name = "<name>"
        request.risk_levels = "<risk_levels>"
        response = client.list_audit_rule_risks(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"
    dbss "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/region"
)

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

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

    client := dbss.NewDbssClient(
        dbss.DbssClientBuilder().
            WithRegion(region.ValueOf("cn-north-4")).
            WithCredential(auth).
            Build())

    request := &model.ListAuditRuleRisksRequest{}
    nameRequest := "<name>"
    request.Name = &nameRequest
    riskLevelsRequest := "<risk_levels>"
    request.RiskLevels = &riskLevelsRequest
```

```
response, err := client.ListAuditRuleRisks(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	成功
400	请求参数错误
403	认证失败
500	服务器内部错误

错误码

请参见[错误码](#)。

3.5.4 查询指定风险规则策略

功能介绍

查询指定风险规则策略

调用方法

请参见[如何调用API](#)。

URI

GET /v1/{project_id}/{instance_id}/dbss/audit/rule/risk/{risk_id}

表 3-220 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
instance_id	是	String	实例ID
risk_id	是	String	风险规则ID

请求参数

表 3-221 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

响应参数

状态码： 200

表 3-222 响应 Body 参数

参数	参数类型	描述
rule_id	String	风险规则ID
rule_name	String	风险名称
status	String	风险规则状态 枚举值： <input type="checkbox"/> OFF <input type="checkbox"/> ON
action	String	操作集合, 中间逗号分隔 LOGIN CREATE_TABLE CREATE_TABLESPACE DROP_TABLE DROP_TABLESPACE DELETE INSERT INSERT_SELECT SELECT SELECT_FOR_UPDATE UPDATE CREATE_USER DROP_USER GRANT OPERATE ALL
schemas	Array of schemas objects	Schema列表
rank	Integer	风险规则优先级
ignore_case	Boolean	是否忽略大小写
risk_level	String	风险级别 枚举值： <input type="checkbox"/> LOW <input type="checkbox"/> MEDIUM <input type="checkbox"/> HIGH <input type="checkbox"/> NO_RISK
db_ids	String	数据库id, 中间逗号分隔 (单个id 小于256位)
execution_symbol	String	执行时长对执行时长阈值的关系 枚举值： <input type="checkbox"/> GREATER <input type="checkbox"/> EQUAL <input type="checkbox"/> LESS <input type="checkbox"/> GREATER_EQUAL <input type="checkbox"/> LESS_EQUAL <input type="checkbox"/> NO_MATCH
execution_time	Integer	设定的执行时长阈值
affect_symbol	String	影响行数对行数阈值的关系： 枚举值： <input type="checkbox"/> GREATER <input type="checkbox"/> EQUAL <input type="checkbox"/> LESS <input type="checkbox"/> GREATER_EQUAL <input type="checkbox"/> LESS_EQUAL <input type="checkbox"/> NO_MATCH
affect_rows	Integer	设定的影响行数阈值

参数	参数类型	描述
client_ips	String	客户端IP段: IP-IP格式, 或IP/XX 格式。各个IP段使用逗号连接

表 3-223 schemas

参数	参数类型	描述
schema	String	schema名称
table	String	表名
column	String	列名

状态码: 400

表 3-224 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-225 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码: 403

表 3-226 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-227 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息。

状态码： 500

表 3-228 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-229 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

请求示例

```
/v1/{project_id}/{instance_id}/dbss/audit/rule/risk/{risk_id}
```

响应示例

状态码： 200

成功

```
{
  "status": "OFF",
  "action": "LOGIN,SELECT,INSERT",
  "schemas": [ {
    "schema": "dbss_audit",
    "table": null,
    "column": null
  } ],
  "rank": 6,
  "ignore_case": false,
  "rule_id": "AWTOHznX7At9UslqwTfm",
  "rule_name": "risk_rule_name_00",
  "risk_level": "MEDIUM",
  "db_ids": "11111,22222",
  "execution_symbol": "GREATER",
  "execution_time": 10000,
  "affect_symbol": "GREATER",
  "affect_rows": 30,
  "client_ips": "192.168.0.1"
}
```

状态码： 400

请求参数错误

```
{
  "error": {
    "error_code": "DBSS.XXXX",
    "error_msg": "XXX"
  }
}
```

状态码： 500

服务器内部错误

```
{
  "error": {
    "error_code": "DBSS.XXXX",
    "error_msg": "XXX"
  }
}
```

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.dbss.v1.region.DbssRegion;
import com.huaweicloud.sdk.dbss.v1.*;
import com.huaweicloud.sdk.dbss.v1.model.*;

public class ShowAuditRuleRiskSolution {

    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");

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

        DbssClient client = DbssClient.newBuilder()
            .withCredential(auth)
            .withRegion(DbssRegion.valueOf("cn-north-4"))
            .build();
        ShowAuditRuleRiskRequest request = new ShowAuditRuleRiskRequest();
        try {
            ShowAuditRuleRiskResponse response = client.showAuditRuleRisk(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

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdbss.v1.region.dbss_region import DbssRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdbss.v1 import *

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

    credentials = BasicCredentials(ak, sk) \

    client = DbssClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DbssRegion.value_of("cn-north-4")) \
        .build()

    try:
        request = ShowAuditRuleRiskRequest()
        response = client.show_audit_rule_risk(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"
    dbss "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/region"
)

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

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

    client := dbss.NewDbssClient(
        dbss.DbssClientBuilder().
            WithRegion(region.ValueOf("cn-north-4")).
```

```
WithCredential(auth).  
Build()  
  
request := &model.ShowAuditRuleRiskRequest{}  
response, err := client.ShowAuditRuleRisk(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	成功
400	请求参数错误
403	认证失败
500	服务器内部错误

错误码

请参见[错误码](#)。

3.5.5 查询隐私数据脱敏规则

功能介绍

查询隐私数据脱敏规则

调用方法

请参见[如何调用API](#)。

URI

GET /v1/{project_id}/{instance_id}/dbss/audit/sensitive/masks

表 3-230 路径参数

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

表 3-231 Query 参数

参数	是否必选	参数类型	描述
offset	否	String	偏移量
limit	否	String	查询记录数

请求参数

表 3-232 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

响应参数

状态码： 200

表 3-233 响应 Body 参数

参数	参数类型	描述
rules	Array of rules objects	规则列表
total	Integer	总数

表 3-234 rules

参数	参数类型	描述
id	String	规则ID
name	String	规则名称
type	String	规则类型
regex	String	规则正则表达式
mask_value	String	替换值
status	String	规则状态
operate_time	String	操作时间

状态码： 400

表 3-235 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-236 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 403

表 3-237 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-238 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

状态码： 500

表 3-239 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 3-240 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码。
error_msg	String	错误请求返回的错误信息。

请求示例

```
/v1/{project_id}/{instance_id}/dbss/audit/sensitive/masks
```

响应示例

状态码： 200

成功

```
{
  "rules": [ {
    "id": "n34W2ngBo47GiyUSKOVl",
    "name": "Passport NO.",
    "type": "BUILD_IN",
    "regex": "-",
    "mask_value": "###",
    "status": "ON",
    "operate_time": "2030-01-01 00:00:06"
  }, {
    "id": "nn4W2ngBo47GiyUSKOVp",
    "name": "Military officer card NO.",
    "type": "BUILD_IN",
    "regex": "-",
    "mask_value": "###",
    "status": "ON",
    "operate_time": "2030-01-01 00:00:05"
  }, {
    "id": "nX4W2ngBo47GiyUSKOU9",
    "name": "Ethnicity",
    "type": "BUILD_IN",
    "regex": "-",
    "mask_value": "###",
    "status": "ON",
    "operate_time": "2030-01-01 00:00:04"
  }, {
    "id": "mn4W2ngBo47GiyUSKOUO",
    "name": "GPS Information",
    "type": "BUILD_IN",
    "regex": "-",
    "mask_value": "###",
    "status": "ON",
    "operate_time": "2030-01-01 00:00:01"
  } ],
  "total": 6
}
```

状态码： 400

请求参数错误

```
{
  "error": {
    "error_code": "DBSS.XXXX",
    "error_msg": "XXX"
  }
}
```

状态码： 500

服务器内部错误

```
{
  "error": {
    "error_code": "DBSS.XXXX",
    "error_msg": "XXX"
  }
}
```


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.dbss.v1.region.DbssRegion;
import com.huaweicloud.sdk.dbss.v1.*;
import com.huaweicloud.sdk.dbss.v1.model.*;

public class ListAuditSensitiveMasksSolution {

    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");

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

        DbssClient client = DbssClient.newBuilder()
            .withCredential(auth)
            .withRegion(DbssRegion.valueOf("cn-north-4"))
            .build();
        ListAuditSensitiveMasksRequest request = new ListAuditSensitiveMasksRequest();
        request.withOffset("<offset>");
        request.withLimit("<limit>");
        try {
            ListAuditSensitiveMasksResponse response = client.listAuditSensitiveMasks(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

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdbss.v1.region.dbss_region import DbssRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdbss.v1 import *

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

risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.

In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment

```
ak = os.getenv("CLOUD_SDK_AK")
sk = os.getenv("CLOUD_SDK_SK")

credentials = BasicCredentials(ak, sk) \

client = DbssClient.new_builder() \
    .with_credentials(credentials) \
    .with_region(DbssRegion.value_of("cn-north-4")) \
    .build()

try:
    request = ListAuditSensitiveMasksRequest()
    request.offset = "<offset>"
    request.limit = "<limit>"
    response = client.list_audit_sensitive_masks(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"
    dbss "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dbss/v1/region"
)

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

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

    client := dbss.NewDbssClient(
        dbss.DbssClientBuilder().
            WithRegion(region.ValueOf("cn-north-4")).
            WithCredential(auth).
            Build())

    request := &model.ListAuditSensitiveMasksRequest{}
    offsetRequest := "<offset>"
    request.Offset = &offsetRequest
    limitRequest := "<limit>"
    request.Limit = &limitRequest
    response, err := client.ListAuditSensitiveMasks(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	成功
400	请求参数错误
403	认证失败
500	服务器内部错误

错误码

请参见[错误码](#)。

4 附录

4.1 状态码

- 正常

返回值	说明
200	请求成功。

- 异常

状态码	编码	说明
400	Bad Request	服务器未能处理请求。
401	Unauthorized	被请求的页面需要用户名和密码。
403	Forbidden	对被请求页面的访问被禁止。
404	Not Found	服务器无法找到被请求的页面。
405	Method Not Allowed	请求中指定的方法不被允许。
406	Not Acceptable	服务器生成的响应无法被客户端所接受。
407	Proxy Authentication Required	用户必须首先使用代理服务器进行验证，这样请求才会被处理。
408	Request Timeout	请求超出了服务器的等待时间。
409	Conflict	由于冲突，请求无法被完成。
500	Internal Server Error	请求未完成，服务异常。
501	Not Implemented	请求未完成，服务器不支持所请求的功能。

状态码	编码	说明
502	Bad Gateway	请求未完成，服务器从上游服务器收到一个无效的响应。
503	Service Unavailable	请求未完成，系统暂时异常。
504	Gateway Timeout	网关超时。

4.2 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	DBSS.10000001	Enter a valid request message	请求消息格式非法	检查参数
400	DBSS.10020101	Enter a valid request message	获取规格列表失败	检查参数
400	DBSS.10020102	Enter a valid request message	操作数据库失败	检查参数
400	DBSS.10020118	Failed to add database, exceeding the limit	添加数据库失败,超出数量限制	删除不需要的数据库或购买新实例
400	DBSS.10020140	Illegal order ID	订单ID不符合要求	检查订单ID
400	DBSS.100210016	Insufficient quota	配额不足	联系管理员
400	DBSS.10020021	Invalid request parameter ID.	请求ID参数不合法	检查参数
401	DBSS.10020100	Failed to authenticate the token in the request	请求所带的Token认证失败	检查token
404	DBSS.10021004	ECS can not found the request page	ECS服务器无法找到被请求页面	检查ecs路径配置

状态码	错误码	错误信息	描述	处理措施
500	DBSS.11000000	Internal system exception. Contact technical support engineers	系统内部异常，请联系技术支持人员	联系管理员

4.3 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。

```
{
  "projects": [
    {
      "domain_id": "65382450e8f64ac0870cd180d14e684b",
      "is_domain": false,
      "parent_id": "65382450e8f64ac0870cd180d14e684b",
      "name": "xxxxxxx",
      "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“我的凭证”。
3. 在“API凭证”页面的项目列表中查看项目ID。

图 4-1 查看项目 ID



5 修订记录

发布日期	修改说明
2024-07-30	第三次正式发布。 添加 错误码 ：DBSS.10020021。
2024-04-10	第二次正式发布。 优化： 包年包月计费模式创建审计实例 补充请求示例内容。
2023-06-16	第一次正式发布。