开放搜索

API参考手册(高级版)V2

API参考手册(高级版)V2

系统为开发者提了一套REST风格的WebAPI服务。本手册将详细介绍WebAPI的功能、使用方法及详细参数的设定。 API说明文档：包括应用相关API，上传数据API，数据搜索API。

API访问上分为公共参数及请求参数两部分，所有请求均必须包含该两部分才能执行。

术语表

术语	全称	中文	说明
app	application	应用	一个应用即一个完整的搜索服务

错误码

API	描述
错误码	返回错误码说明

公共调用方式

API	描述
公共参数	公共调用及返回参数说明
授权机制	详细说明验签过程

应用操作接口

API	描述
应用管理	查看应用
数据处理	上传、修改、删除文档
搜索	查询
错误信息	查看错误日志
索引重建	创建、查看索引重建任务

- API版本：V2
- 最近更新时间：2014-01-01

发布时间	更新说明	当前状态
2014-01-01	发布V2版本，优化使用接口，提高查询性能	stable
2013-01-01	发布V1版本，提供主流系统操作api接口	不维护状态，尽快升级V2

调用方式

服务地址

OpenSearch为分区域部署，每个区域访问地址均不同，具体服务地址请查看访问应用-》基本详情中的API入口部分获取，切勿随便使用。

通信协议

支持HTTP协议

请求方法

推送数据建议采用POST方式，搜索建议采用GET方式。 注意：使用POST方式向API提交数据时需要将HTTP请求的“Content-Type”设置为“application/x-www-form-urlencoded”，否则会导致API解析失败。

请求参数

每个操作都需要包含公共请求参数及具体请求所特有的请求参数。

字符编码

请求及返回结果都仅支持UTF-8字符集。

公共请求参数

公共请求参数是指**每个接口**都需要使用到的请求参数。

名称	类型	是否必须	描述
Version	String	是	API 版本，当前版本为：v2
AccessKeyId	String	是	阿里云颁发给用户的访问服务所用的密钥ID
Signature	String	是	签名结果串，关于签名的计算方法，请参见**授权机制**。
SignatureMethod	String	是	签名方式，目前支持HMAC-SHA1
Timestamp	String	是	请求的时间戳。日期格式按照ISO8601标准表示，**必须使用UTC时间**。格式为YYYY-MM-DDThh:mm:ssZ 例如，2014-05-26T12:00:00Z（为北京时间2014年5月26日20点0分0秒）
SignatureVersion	String	是	签名算法版本，目前版本是1.0
SignatureNonce	String	是	唯一随机数，用于防止网络重放攻击。用户在不同请求间要使用不同的随机数值，建议使用13位毫秒时间戳+4位随机数

示例

```
http://$host/index/doc/$app_name?Version=v2&AccessKeyId=$accessKeyId&Signature=$signature&SignatureMethod=HMAC-SHA1&Timestamp=$timestamp&SignatureVersion=1.0&SignatureNonce=$signatureNonce
```

公共返回结果

用户发送的每次接口调用请求，无论成功与否，系统都会返回一个status字段给用户，用来表示本次请求的正确与否。同时，错误的请求将会返回错误代码及错误描述，供用户调试。

请求结束后请务必检查返回值是否正常，并根据错误信息及时修改，否则可能出现数据丢失或者请求无结果的情况。

示例

JSON示例

返回成功：

```
{
   "status": "OK"
}
```

返回错误：

```
{
   "status":"FAIL",
   "errors":[
      {"code":2001, "message":"App is not found"}
   ]
}
```

授权机制

OpenSearch服务会对每个访问的请求进行身份验证，通过使用Access Key ID和Access Key Secret进行对称加密的方法来验证请求的发送者身份。Access Key ID和Access Key Secret由阿里云官方颁发给访问者（可以通过阿里云官方网站申请和管理），其中Access Key ID用于标识访问者的身份；

Access Key Secret是用于加密签名字符串和服务器端验证签名字符串的密钥，必须严格保密，只有阿里云和用户知道。

【特别注意】

标准版应用必须要使用实现了 V3 版 API 的SDK，或使用我们提供的已实现V3 版API的 SDK 才能正常访问，若使用实现了 V2版API 或 实现了V2版 API的 SDK访问标准版应用会报错（错误码： 2030 ，报错提示信息：Request is not permitted）。

签名步骤

1. 使用请求参数构造规范化的请求字符串（Canonicalized Query String）

a. 按照参数名称的字典顺序对请求中所有的请求参数（包括文档中描述的“公共请求参数”和给定了的请求接口的自定义参数，但不能包括“公共请求参数”中提到Signature参数本身）进行排序；

注：当使用GET方法提交请求时，这些参数就是请求URI中的参数部分（即URI中“?”之后由“&”连接的部分）；

b. 对每个请求参数的名称和值分别进行URL编码（仅支持UTF-8字符集）’’’ 。URL编码的编码规则是：

- 对于字符 A-Z、a-z、0-9以及字符“-”、“_”、“.”、“~”不编码；
- 对于其他字符编码成“%XY”的格式，其中XY是字符对应ASCII码的16进制表示。比如英文的双引号（” ）对应的编码就是%22；
- 对于扩展的UTF-8字符，编码成“%XY%ZA…”的格式；
- 需要说明的是英文空格（ ）要被编码是%20，而不是加号（+）。
- **注意：一般支持URL编码的库（比如Java中的java.net.URLEncoder）都是按照“application/x-www-form-urlencoded”的MIME类型的规则进行编码的。实现时可以直接使用这类方式进行编码，把编码后的字符串中加号（+）替换成%20、星号（*）替换成%2A、%7E替换回波浪号（~），即可得到上述规则描述的编码字符串。**

c. 将编码后的参数名称和值使用英文等号（=）连接，得到若干参数对；

d. 将参数对按照原参数名称字典序排序结果使用&符号依次连接，即得到规范化请求字符串。

2. 使用上一步构造的规范化字符串按照下面的规则构造用于计算签名的字符串

生成方式如下：

```
StringToSign=
HTTPMethod + "&" +
percentEncode("/") + "&" +
percentEncode(CanonicalizedQueryString)
```

- 其中HTTPMethod是提交请求用的HTTP方法，比如GET。
- percentEncode(“/”)是按照1.b中描述的URL编码规则对字符“/”进行编码得到的值，即“%2F”。
- percentEncode(CanonicalizedQueryString)是对第1步中构造的规范化请求字符串按1.b中描述的URL编码规则编码后得到的字符串。

3. 按照RFC2104的定义，使用上面的用于签名的字符串计算签名HMAC值。

注意：计算签名时使用的Key就是用户持有的Access Key Secret并加上一个“&”字符(ASCII:38)，使用的哈希算法是SHA1。

4. 按照Base64编码规则把上面的HMAC值编码成字符串，即得到签名值（Signature）。

5. 最后将得到的签名值作为Signature参数添加到请求参数中，即完成对请求签名的过程。

注意: 得到的签名值在作为最后的请求参数值提交给OPENSEARCH服务器的时候，要和其他参数一样，按照RFC3986的规则进行URL编码)

示例

以搜索请求为例，签名前的请求url原串（未urlencode的）为：

公共参数
：Version=v2&AccessKeyId=testid&SignatureMethod=HMAC - SHA1&SignatureVersion=1.0&SignatureNonce=14053016951271226&Timestamp=2014-07-14T01:34:55Z
请求参数为：query=config=format:json,start:0,hit:20&&query=default:'的'&index_name=ut_3885312&format=json&fetch_fields=title;gmt_modified

经过第一步排序、编码及拼接等工作，得到Canonicalized Query String为：

AccessKeyId=testid&SignatureMethod=HMAC-SHA1&SignatureNonce=14053016951271226&SignatureVersion=1.0&Timestamp=2014-07-14T01%3A34%3A55Z&Version=v2&fetch_fields=title%3Bgmt_modified&format=json&index_name=ut_3885312&query=config%3Dformat%3Ajson%2Cstart%3A0%2Chit%3A20&&query%3Ddefault%3A%27%E7%9A%84%27

搜索使用GET协议，那么第二步的StringToSign就是：

GET&%2F&AccessKeyId%3Dtestid&SignatureMethod%3DHMAC-SHA1&SignatureNonce%3D14053016951271226&SignatureVersion%3D1.0&Timestamp%3D2014-07-14T01%253A34%253A55Z&Version%3Dv2&fetch_fields%3Dtitle%253Bgmt_modified&format%3Djson&index_name%3Dut_3885312&query%3Dconfig%253Dformat%253Ajson%252Cstart%253A0%252Chit%253A20%2526%2526query%253Ddefault%253A%2527%25E7%259A%2584%2527

假如使用的Access Key Id是“testid”，Access Key Secret是“testsecret”，用于计算HMAC的Key就是“testsecret&”，则计算得到的签名值Signature是：

AXA41Uk1UbIyLDttENNn34mqRbE=

签名后的请求URL为（注意增加了Signature参数）：

http://$host/search?query=config%3Dformat%3Ajson%2Cstart%3A0%2Chit%3A20&&query%3Ddefault%3A%27%E7%9A%84%27&index_name=ut_3885312&format=json&fetch_fields=title%3Bgmt_modified&Version=v2&AccessKeyId=testid&SignatureMethod=HMAC-SHA1&SignatureVersion=1.0&SignatureNonce=14053016951271226&Timestamp=2014-07-14T01%3A34%3A55Z&Signature=AXA41Uk1UbIyLDttENNn34mqRbE%3D

错误码

请求出现错误后，将会有错误码及错误信息返回，您可以在这里进一步确定错误原因，及时修改，确保访问正确进行。

常见错误处理

错误码	处理方式
1000	一般为语法或者超时引起，如果多次刷新不再出现，则是超时引起，如果仍出现，则语法有问题，请对照文档仔细检查，如分隔符、函数字段类型等
2112	排序表达式中的text_relevance(field)、fieldterm_proximity(field)等文本feature中的field必须在查询的索引包含的源字段中，否则会报错，但不影响搜索结果。
3007	对于API推送系统是有频率限制，请控制好频率重试
4003	可以先按照文档样例，试下签名结果是否一致，判断是否是签名算法问题。如果不是，请检查下参数按照字典序排序后应该是公共参数（大写字母）在前，请求参数（小写字母）在后。另外还有空格等一些编码规则，具体参考授权文档介绍
4007	一般Json字段内容中包含双引号或者不可见字符会导致格式解析失败，请转义或者过滤后重试
4010	TimeStamp参数是有过期时间的，请按照要求格式取当前时间来计算
5001	没有找到对应的用户，一般为ACCESSKEY信息不正确，或者使用区域域名错误（API域名请以应用管理-》基本信息-》API入口为准），请检查修改后重试
5008	服务内部是通过Accesskey来进行用户身份校验的，请确保AccessKey已经开启，您可以通过控制台AccessKey管理入口来创建和删除
6013	start+hit不能超过5000，否则会报错无结果。需要超过5000的请求，请查看下API文档中的SCROLL接口，看是否满足需求
6015	请及时到控制台配额管理处进行QPS峰值的调整，否则超过的请求会被丢弃
6127	除了query子句，其他子句出现的字段都必须配置为属性字段才能使用。请修改应用结构后重试

系统级别（1000-1999）

错误码	错误说明
1000	系统内部错误
1001	没有找到模版
1003	不支持的索引类型
1004	服务暂时不可用，请稍后再试

应用相关（2000-2999）

错误码	错误说明
2001	待查应用不存在
2002	应用已经存在
2003	到达创建应用总限制
2004	应用名不可用。应用名由数字、26个英文字母或下划线组成，长度不超过30位
2005	应用名称没有设定
2006	新应用名称没有设定
2007	备注不超300字
2008	摘要配置参数错误
2009	更新状态失败
2010	应用暂停中
2011	应用冻结中
2012	应用未开启
2013	删除失败，没有此应用
2014	文件上传失败
2016	区域信息没有
2017	此应用并不属于当前区域
2099	当前接口暂时不提供服务。
2101	表达式不存在
2102	表达式名称被占用
2103	到达该应用表达式总数限制
2104	表达式名不可用。表达式名由数字、26个英文字母或下划线组成，长度不超过30位

2105	表达式名称没有设定
2106	新表达式名称没有设定
2107	表达式备注不超过300字
2108	表达式备注格式错误
2109	表达式格式错误
2110	表达式长度超过限制
2111	表达式id未指定
2112	表达式错误
2113	表达式不能为空
2114	操作错误
2201	粗排配置名没有设定
2202	粗排配置名已经存在
2203	粗排配置个数超出限制
2204	粗排配置名错误。只能由数字、26个英文字母或下划线组成
2205	粗排配置名长度超出限制
2206	粗排字段必须是数值型
2207	粗排配置不存在
2208	粗排配置错误，必须包含字段
2209	粗排配置权重错误,必须是-100000到100000之间的非0数值，浮点数精度支持6位
2210	与系统默认粗排配置重名
2211	timeliness()的参数必须是INT类型
2112	排序表达式错误
2551	查询指定的下拉提示规则不存在

文档相关（3000-3999）

错误码	**错误说明**
3001	文档不能为空
3002	文档大小超过限制
3003	已经到最大文档数
3004	保存文档失败
3005	doc格式错误

3006	文档操作cmd不合法
3007	请求过于频繁
3008	文档总长度太长
3009	没有文档id

授权相关（4000-4999）

错误码	错误说明
4001	认证失败
4002	需要设置签名
4003	签名验证失败
4004	需要设置SignatureNonce
4005	SignatureNonce不能重复使用
4006	SignatureNonce验证失败
4007	解析JSON格式失败
4008	用户名称不能为空，请检查域名正确性
4009	需要指定用户标识
4010	时间过期
4011	demo帐号禁止执行的操作
4012	数据表不存在
4013	Timestamp格式错误
4014	需要设置Timestamp

用户相关（5000-5999）

错误码	错误说明
5001	用户不存在
5002	用户名不正确
5003	需要用户登录
5005	用户未开通OpenSearch服务，请前往阿里云官网开通
5008	用户没有启用ACCESSKEY
5100	用户没有此区域的操作权限
5004	用户未缴费

5005	用户未开通OpenSearch服务，请前往阿里云官网开通
5006	欠费冻结中
5008	用户没有启用ACCESSKEY
5009	用户已经删除
5010	ACCESSKEY 已经禁用
5011	通过邮箱获取到多个用户
5012	CODE_USER_ALIYUN_USER_ID_INVALID，错误信息为空
5013	CODE_USER_ALIYUN_BID_INVALID，错误信息为空
5014	CODE_USER_CLIENT_ID_INVALID，错误信息为空
5015	CODE_USER_ID_INVALID，错误信息为空
5100	用户没有此区域的操作权限

搜索相关（6000-6999）

错误码	错误说明
6001	查询query为空
6002	并不被支持的搜索key关键字
6003	并不被支持的搜索field关键字
6004	复杂查询为空
6005	field无效
6006	请求包含太多应用名
6007	超出多索引查询每个模板中索引总数
6008	请求串语法错误，解析失败
6009	查询子句过长
6010	无效的rerank size
6011	SignatureNonce格式错误
6013	start+hit超过系统限制
6014	因系统繁忙，请求被丢弃
6015	因流量超出配额，请求被丢弃
6016	查询hit数超过系统限制
6017	目前scroll只支持search_type为scan，也就是说

	设置了参数scroll，就必须设置参数search_type=scan
6018	设置了scroll参数，但没有search_type参数
6019	传入的scroll_id参数解析失败
6020	无效的scroll参数值
6021	scroll请求不支持Aggregate/Sort/Distinct，当传入这些clause时，会报错
6022	scroll_id已经过期失效了
6100	查询词为空
6101	查询的索引字段不存在
6102	Query中的数值范围错误
6103	Filter中的表达式返回值必须为bool类型
6104	Sort中的表达式返回值不能为bool类型
6105	Sort中存在相同的表达式
6106	查询query语句非法
6107	统计函数表达式的返回值不能为bool或者string类型
6108	统计中的范围必须为升序
6109	统计中的范围表达式返回值类型错误
6110	统计函数不存在
6111	不支持的统计函数
6112	Query 子句错误
6113	Filter子句错误
6114	Aggregate子句错误
6115	Sort子句错误
6116	Distinct子句错误
6117	查询中包含未知的子句
6118	语法错误
6119	Distinct子句中的dist_count值错误，应该为大于0的整数
6120	Distinct子句中的dist_times值错误，应该为大于0的整数
6121	Distinct子句中的reserved值错误，应为true/false
6122	Distinct子句缺少distinct_key
6123	Distinct子句中的grade值错误，例如为空，或非

	数值
6124	Distinct子句中包含distinct个数不对,个数应在(0,2]
6125	Distinct子句中的max_item_count值错误，应该为大于0的整数
6126	Distinct子句中的update_total_hit值错误，应为true/false
6127	请求中包含了未定义的attribute字段
6128	表达式中的二元操作符的两边的表达式结果类型不匹配
6129	表达式中的二元操作符的两边表达式不能同时为常量
6130	二元逻辑运算表达式类型错误，应为bool类型
6131	二元表达式中不支持string类型
6132	二元表达式中不支持数组类型
6133	位操作中的类型错误
6134	常量表达式的返回值类型错误
6300	常量表达式类型应是整数或浮点数
6301	位取反操作数类型必须为整数
6302	取负数操作数必须为数值
6303	逻辑非操作数必须为数值
6304	二元运算操作数类型错误
6305	非法的二元运算符
6306	函数参数类型错误
6307	函数未定义
6308	函数参数个数错误
6309	非法的数组操作
6310	可过滤字段不存在
6311	数组字段被错当作单值使用
6312	单值字段被错当作数组使用
6313	数组字段下标越界(小于0)
6314	不支持的字段类型
6315	索引字段参数不存在
6316	Query中没有指定索引
6317	Filter子句中只能使用一次公式

6318	公式语法解析出错
6500	搜索语法中包含不存在的字段
6501	在线系统没有索引数据
6502	用户query语法错误
6601	一个索引字段只能包含在一个规则中
6602	没有查询词，如default:’’ 的情况
6603	查询中的索引字段没有在查询分析规则中指定
6604	关键词没有使用引号括起来，如default:xxx，正确为default:’xxx’
6605	双引号查询不能配置查询分析规则
6607	disable参数格式错误
6608	disable指定关闭的索引字段不存在
6609	disable指定关闭的功能列表不存在
6610	查询分析后的query为空（原query为空，或者全部是stopword）
6611	查询中没有指定索引字段
6001	查询query为空
6002	并不被支持的搜索key关键字
6003	并不被支持的搜索field关键字
6004	复杂查询为空
6005	field无效
6006	请求包含太多应用名
6007	超出多索引查询每个模板中索引总数
6008	请求串语法错误，解析失败
6009	查询子句过长
6010	无效的rerank size
6011	SignatureNonce格式错误
6013	start+hit超过系统限制
6014	因系统繁忙，请求被丢弃
6015	因流量超出配额，请求被丢弃
6016	查询hit数超过系统限制
6017	目前scroll只支持search_type为scan，也就是说设置了参数scroll，就必须设置参数search_type=scan
6018	设置了scroll参数，但没有search_type参数

6019	传入的scroll_id参数解析失败
6020	无效的scroll参数值
6021	scroll请求不支持Aggregate/Sort/Distinct，当传入这些clause时，会报错
6022	scroll_id已经过期失效了
6100	查询词为空
6101	查询的索引字段不存在
6102	Query中的数值范围错误
6103	Filter中的表达式返回值必须为bool类型
6104	Sort中的表达式返回值不能为bool类型
6105	Sort中存在相同的表达式
6106	查询query语句非法
6107	统计函数表达式的返回值不能为bool或者string类型
6108	统计中的范围必须为升序
6109	统计中的范围表达式返回值类型错误
6110	统计函数不存在
6111	不支持的统计函数
6112	Query 子句错误
6113	Filter子句错误
6114	Aggregate子句错误
6115	Sort子句错误
6116	Distinct子句错误
6117	查询中包含未知的子句
6118	语法错误
6119	Distinct子句中的dist_count值错误，应该为大于0的整数
6120	Distinct子句中的dist_times值错误，应该为大于0的整数
6121	Distinct子句中的reserved值错误，应为true/false
6122	Distinct子句缺少distinct_key
6123	Distinct子句中的grade值错误，例如为空，或非数值
6124	Distinct子句中包含distinct个数不对,个数应在(0,2]

6125	Distinct子句中的max_item_count值错误，应该为大于0的整数
6126	Distinct子句中的update_total_hit值错误，应为true/false
6127	请求中包含了未定义的attribute字段
6128	表达式中的二元操作符的两边的表达式结果类型不匹配
6129	表达式中的二元操作符的两边表达式不能同时为常量
6130	二元逻辑运算表达式类型错误，应为bool类型
6131	二元表达式中不支持string类型
6132	二元表达式中不支持数组类型
6133	位操作中的类型错误
6134	常量表达式的返回值类型错误
6300	常量表达式类型应是整数或浮点数
6301	位取反操作数类型必须为整数
6302	取负数操作数必须为数值
6303	逻辑非操作数必须为数值
6304	二元运算操作数类型错误
6305	非法的二元运算符
6306	函数参数类型错误
6307	函数未定义
6308	函数参数个数错误
6309	非法的数组操作
6310	可过滤字段不存在
6311	数组字段被错当作单值使用
6312	单值字段被错当作数组使用
6313	数组字段下标越界(小于0)
6314	不支持的字段类型
6315	索引字段参数不存在
6316	Query中没有指定索引
6317	Filter子句中只能使用一次公式
6318	公式语法解析出错
6500	搜索语法中包含不存在的字段
6501	在线系统没有索引数据

6502	用户query语法错误
6601	一个索引字段只能包含在一个规则中
6602	没有查询词，如default:’ ’ 的情况
6603	查询中的索引字段没有在查询分析规则中指定
6604	关键词没有使用引号括起来，如default:xxx，正确为default:’ xxx’
6605	双引号查询不能配置查询分析规则
6607	disable参数格式错误
6608	disable指定关闭的索引字段不存在
6609	disable指定关闭的功能列表不存在
6610	查询分析后的query为空（原query为空，或者全部是stopword）
6611	查询中没有指定索引字段

数据处理相关（7000-7999）

错误码	错误说明
7100	没有错误发生
7101	单个文档过长
7102	文档所属应用的元信息错误（clientid、应用名或表名等不正确）
7103	HA3 文档格式错误: 字段解析失败
7104	JSON文档格式错误：字段解析失败
7105	JSON 文档格式错误: json非法
7106	JSON 文档格式错误: json非法
7107	不支持的编码
7108	编码转换失败
7109	fields中没有id字段
7110	fields中id定义不合法
7111	fields中包含保留字段
7201	HA3 文档格式错误: cmd 非法(cmd 非 ADD/UPDATE/DELETE)
7202	JSON 文档格式错误: cmd 非法(cmd 非 ADD/UPDATE/DELETE)
7301	主键字段不存在

7302	字段数据类型错误
7303	数组字段相关错误
7401	文档总数超出配额
7402	每日更新文档数超出配额
7403	单次导入的数据大小超出配额
7500	系统内部错误
7501	云梯Hive待同步字段的列号超出了当前数据的列数范围
7502	从Mysql中读取到的主键字段为空,请联系数据库管理员
7503	JsonKeyValueExtractor内容转换错误: Json格式非法
7504	JsonKeyValueExtractor内容转换错误: key不存在
7505	TairLDBExtractor内容转换错误: namespace非法（应为int32类型）
7506	TairLDBExtractor内容转换错误: 从Tair中读取数据失败
7507	MySql实时同步过滤条件格式错误
7508	系统内部错误: 内容转换插件初始化失败
7509	TairLDBExtractor内容转换配置错误：Tair连接失败，请检查configId 或 namespace 是否有效
7510	KVExtractor内容解析错误：KV格式无法解析
7511	OSS 数据读取失败
7512	OSS 内容长度超过限度
7513	OSS 内容解析错误
7514	系统内部错误: OSS LOG 格式不兼容
7515	过滤条件执行错误
7516	字段映射过程中源表字段缺失
7517	StringCatenateExtractor内容转换错误: 源字段不存在
7518	StringCatenateExtractor内容转换错误: 不支持多值字段
7601	任务执行错误
7100	没有错误发生
7101	单个文档过长
7102	文档所属应用的元信息错误（clientid、应用名或表名等不正确）

7103	HA3 文档格式错误: 字段解析失败
7104	JSON文档格式错误：字段解析失败
7105	JSON 文档格式错误: json非法
7106	JSON 文档格式错误: json非法
7107	不支持的编码
7108	编码转换失败
7109	fields中没有id字段
7110	fields中id定义不合法
7111	fields中包含保留字段
7201	HA3 文档格式错误: cmd 非法(cmd 非 ADD/UPDATE/DELETE)
7202	JSON 文档格式错误: cmd 非法(cmd 非 ADD/UPDATE/DELETE)
7301	主键字段不存在
7302	字段数据类型错误
7303	数组字段相关错误
7401	文档总数超出配额
7402	每日更新文档数超出配额
7403	单次导入的数据大小超出配额
7500	系统内部错误
7501	云梯Hive待同步字段的列号超出了当前数据的列数范围
7502	从Mysql中读取到的主键字段为空,请联系数据库管理员
7503	JsonKeyValueExtractor内容转换错误: Json格式非法
7504	JsonKeyValueExtractor内容转换错误: key不存在
7505	TairLDBExtractor内容转换错误: namespace非法（应为int32类型）
7506	TairLDBExtractor内容转换错误: 从Tair中读取数据失败
7507	MySql实时同步过滤条件格式错误
7508	系统内部错误: 内容转换插件初始化失败
7509	TairLDBExtractor内容转换配置错误：Tair连接失败，请检查configId 或 namespace 是否有效
7510	KVExtractor内容解析错误：KV格式无法解析
7511	OSS 数据读取失败

7512	OSS 内容长度超过限度
7513	OSS 内容解析错误
7514	系统内部错误: OSS LOG 格式不兼容
7515	过滤条件执行错误
7516	字段映射过程中源表字段缺失
7517	StringCatenateExtractor内容转换错误: 源字段不存在
7518	StringCatenateExtractor内容转换错误: 不支持多值字段
7519	数据源配置中，表名模式配置错误
7520	文档内容长度超过限制
7600	系统内部模板错误
7601	任务执行错误
7602	更新app失败
7701	数据清理任务错误：指定过滤字段不存在
7801	文档格式错误

文档错误内部通知（8000-8999）

错误码	错误说明
8001	保存错误信息失败
8002	必要参数缺失
8003	应用不存在
8004	参数错误
8001	保存错误信息失败
8002	必要参数缺失
8003	应用不存在
8004	参数错误

模板相关（9000-9999）

错误码	错误说明
9001	用户名为空
9002	应用名为空

9003	模板名不可用。模板名只能由数字、26个英文字母或下划线组成
9004	模板名长度不可超过30位
9005	查询模板信息出错
9006	模板名字已存在
9007	插入模板信息出错
9008	无效的数据
9009	定义的字段数目超过系统允许的最大字段数
9010	此字段保留字段名
9011	字段已存在
9012	索引名称必须以字母开头，由数字、26个英文字母或下划线组成，长度不超过30位，多值字段类型不能为SWS_TEXT或TEXT
9013	不支持数组
9014	不支持主键
9015	未设定主键
9016	主键不唯一
9017	更新信息失败
9018	删除信息失败
9019	包含多个索引字段的搜索字段最多4个
9020	同一个STRING/TEXT类型的索引字段不能进入多个只包含一个字段的搜索字段中
9021	索引名称必须以字母开头，由数字、26个英文字母或下划线组成，长度不超过30个
9022	该表已经关联
9023	索引名不能包含多类型的字段
9100	系统内部错误
9101	该字段超过数量限制
9102	该数据源未被用到
9103	无效的外表连接
9104	最多2级关联
9105	待查模板不存在
9501	用户名为空
9502	应用名为空
9519	未指定模板

9600	系统内部错误
9902	插件字段类型错误
9999	此域名不提供本服务
9001	用户名为空
9002	应用名为空
9003	模板名不可用。模板名只能由数字、26个英文字母或下划线组成
9004	模板名长度不可超过30位
9005	查询模板信息出错
9006	模板名字已存在
9007	插入模板信息出错
9008	无效的数据
9009	定义的字段数目超过系统允许的最大字段数
9010	此字段保留字段名
9011	字段已存在
9012	索引名称必须以字母开头，由数字、26个英文字母或下划线组成，长度不超过30位，多值字段类型不能为SWS_TEXT或TEXT
9013	不支持数组
9014	不支持主键
9015	未设定主键
9016	主键不唯一
9017	更新信息失败
9018	删除信息失败
9019	包含多个索引字段的搜索字段最多4个
9020	同一个STRING/TEXT类型的索引字段不能进入多个只包含一个字段的搜索字段中

数据同步相关（10000-）

错误码	**错误说明**
10001	没有指定的tddl group key，tddl信息获取失败
10002	获取字段失败或者表不存在
10011	连接agg失败
10012	应用里存在doc

10013	应用不是自定义结构
10110	该任务已结束
10010	部分数据源有问题，已经忽略有错误的数据
10014	数据源类型错误
10100	创建任务失败，未结束的任务已经存在
10101	没有指定应用ID
10106	没有指定应用ID
10107	没有指定应用ID
10102	ACTION无效
10112	文档数量超过限制
10201	获取配额列表失败
10202	更新配额失败
10301	参数错误：参数未提供或者格式不正确
10302	时间参数错误
10303	数据源未配置
10304	该表配额超限
10305	OSS参数错误
10306	OSS BUCKET名称无效
10307	OSS 记录类型无效
10308	OSS BUCKET日志功能未开启
10309	存在未完成的任务
10310	不是运行中的应用，无法创建任务
10311	时间范围不合法
10312	应用描述长度超过限制，最多600字
10313	OSS 内容格式不合法
10314	OSS BUCKET所在区域ACL网络不通
10315	OSS BUCKET的地址信息不合法
10330	数据源参数不合法
10350	连接ODPS服务失败
10351	ODPS 返回错误
10400	OSS前缀不合法
10450	字段不存在

应用操作接口

查看应用信息

查看一个应用信息

URL

/index/$app_name （ app_name为要创建的应用名称 ）

支持格式

JSON

HTTP请求方式

GET、POST

请求参数

参数	类型	必需	取值范围	默认值	描述
action	string	是	status		status

返回结果

参数	类型	描述
status	string	访问结果，OK为成功，FAIL为失败，请根据返回错误码进行排查
result	string	应用信息，具体信息如下说明

- index_name：应用名
- pv：每日搜索请求数
- doc_last_update_time：上次更新时间
- quota：是否超配额信息
- fields：应用结构信息

- from_table：表结构信息

示例

请求：（此处省略了公共参数及编码等因素）

```
http://$host/index/test_create_index?action=status
```

成功返回：


```
{"result": {
"index_name": "test_create_index",
"pv": 0,
"doc_last_update_time": 0,
"total_doc_num": 10000,
"fields": {
"from_table": {
"main": {
"id": {
"name": "id",
"type": "TEXT",
"is_multi": "1",
"is_pk": "0"
}
}
},
"join_map": [],
"master": "main",
"level": {
"main": "0"
}
},
"merge_table": {
"id": {
"name": "id",
"type": "LITERAL",
"is_pk": "1",
"is_multi": 0
}
},
"primary_key": "id"
},
"indexes": {
"search_fields": {
"id": {
"fields": [
"id"
]
}
},
"filter_fields": [
"id",
```

```
"int_arr",
"literal_arr"
]
}
},
"errors":[],
"status": "OK",
"request_id": "142226700200407760049589 1"}
```

错误返回：

```
{"result":null,
"status":"FAIL",
"errors":[{"code":2001,"message":"App is not found"}],
"RequestId":"1422266985017103400 3366"
}
```

获取应用列表

获得当前应用

URL

/index

支持格式

JSON

HTTP请求方式

GET、POST

请求参数

参数	类型	必需	取值范围	默认值	描述
page	int	否			获取第几页应用列表
page_size	int	否			每页返回的应用个数

返回结果

参数	类型	描述

status	string	访问结果，OK为成功，FAIL为失败，请根据返回错误码进行排查
created	int	应用创建时间戳
name	string	应用名称
id	int	应用id
total	int	总的应用个数

示例

请求：（此处省略了公共参数及编码等因素）

```
http://$host/index?page=1&page_size=1
```

成功返回：

```
{
"result": [
{
"id": "107557",
"name": "test_create_index",
"description": "test",
"created": "1422266999"
}
],
"total": "2",
"status": "OK",
"RequestId": "1422268539021275800907873"
}
```

上传文档

支持新增、更新、删除的批量操作

URL

/index/doc/$app_name（app_name为要操作的应用名称）

支持格式

JSON

HTTP请求方式

POST

请求参数

参数	类型	必需	取值范围	默认值	描述
action	string	是			push
table_name	string	是			要上传数据的表名
items	string	是			规定JSON格式，如下所示

items格式：

```
[
{
"cmd": "add",
"timestamp": 1401342874777,
"fields": {
"id": "1",
"title": "This is the title",
"body": "This is the body"
}
},
{
"cmd": "update",
"timestamp": 1401342874778,
"fields": {
"id": "2",
"title": "This is the new title"
}
},
{
"cmd": "delete",
"fields": {
"id": "3"
}
}
]
```

- cmd : 必选字段。定义该文档的操作行为，可以为“add”、“update”、“delete”。建议一个请求中进行批量更新操作，提高网络交互及处理效率。“add”表示新增文档，如果该主键对应文档已经存在，则执行先“delete”再“add”的操作；“update”表示更新文档，对该主键对应文档进行部分字段更新，如果未存在主键文档，则执行“add”操作；“delete”表示删除文档，如果该主键对应文档已经不存在，则认为删除成功。
- timestamp : 可选字段。用来记录文档实际发生时间，单位为毫秒。系统会用该时间戳来作为同一主

键文档更新顺序的判断标准。如果没有timestamp项，则默认以文档发送到OpenSearch的时间作为文档更新时间来进行保序操作。
- fields：必选字段。要操作的文档内容，主键字段必选，系统所有操作都是通过主键来进行的。对于“delete”只需要提供文档主键即可。
- 对于Array类型，需要使用JsonArray来处理，如[{“fields”:{ “id”: “0”,” int_array”: [14,85],” string_array”:[“abc”,” xyz”]},” cmd”: “ADD”}]；
- 注意：最外层是JsonArray类型，支持多个文档批量操作。

返回结果

参数	类型	描述
status	string	执行结果，OK为成功，FAIL为失败，请根据返回错误码进行排查
request_id	string	该条查询的记录id，主要用于排查问题使用

示例

请求：（此处省略了公共参数及编码等因素）

```
http://$host/index/doc/test_create_index?action=push&table_name=main

//items建议放到body体中
items=[{"cmd":"add","timestamp": 1401342874777,"fields": {"id": "1","title": "This is the title","body": "This is the body"}]
```

成功返回：

```
{"status":"OK","request_id":"142234864206580510037358 7"}
```

错误返回：

```
{"status":"FAIL","errors":[{"code":4012,"message":"Table dose not exist"}],"request_id":"142234873908422230023407 2"}
```

注意事项

- 使用API/SDK推送数据有次数及大小限制，具体值请参考系统限制项。
- 数据上传后请务必检查返回值，并对相关错误码进行重试（尤其是3007错误），否则会出现数据丢失情况。同时，数据处理是异步的，系统返回“OK”后只表示系统接收数据成功，数据处理过程的错误会在控制台错误信息中展示，请注意及时检查。
- post的数据大小有限制，如果您上传的文档过大（2M以上），服务器将拒绝接收任何参数，同时返

回异常。
- POST的url及body部分最好都要做url_encode，否则会出现解析及签名问题。

搜索类

搜索

系统提供了丰富的搜索语法以满足用户各种场景下的搜索需求。

URL

/search

支持格式

JSON

HTTP请求方式

GET

请求参数

参数	类型	必需	取值范围	默认值	描述
query	string	是			搜索主体，包含所有的查询条件。主要子句分别包含config子句、query子句、sort子句、filter子句、aggregate子句、distinct子句、kvpairs子句。
index_name	string	是			要查询的应用名。支持同一查询条件下的多应用同时查询，应用名间使用英文分

					号（;）分隔。
fetch_fields	string	否		全部“可展示”字段。	可以通过此参数获取本次查询需要的字段内容，多个字段使用英文分号（;）分隔
qp	string	否		已上线规则	指定要使用的查询分析规则，多个规则使用英文逗号（,）分隔
disable	string	否			关闭已生效的查询分析功能
first_formula_name	string	否		系统中默认粗排表达式名字	设置粗排表达式名字
formula_name	string	否		系统中默认精排表达式名字	设置精排表达式名字
summary	string	否		获取系统结果摘要配置	动态摘要的配置，可以指定某些字段的飘红、截断等操作。

- query参数：查询的主体。可以通过若干子句的组合来实现多样的搜索需求，其中query子句为必选，子句与子句之前通过“&&”进行连接；
- disable参数：目前仅支持”qp”。
- index_name参数：多应用联合查询说明请见本节的“多应用联合查询”介绍；
- fetch_fields参数：返回文本数据大小对查询性能影响较大，建议只获取需要的字段。
- summary参数：格式如下表，summary_element_prefix与summary_element_postfix必须同时设定；同时summary_element与（summary_element_prefix、summary_element_postfix）是相互影响的，出现在后面的配置会覆盖前面。另外，目前不支持摘要和飘红单独设置。

参数	类型	必需	取值范围	默认值	描述
summary_field	string	是			要做摘要的字段
summary_element	string	否		em	飘红标签，html标签去掉左右尖括号
summary_ell	string	否		...	摘要的结尾

ipsis					省略符
summary_snipped	int	否		1	选取的摘要片段个数
summary_len	string	否			摘要要展示的片段长度
summary_element_prefix	string	否			飘红的前缀，必须是完整的html标签，如
summary_element_postfix	string	否			飘红的后缀，必须是完整的html标签，如

返回结果

参数	类型	描述
status	string	执行结果，OK为成功，FAIL为失败，请根据返回错误码进行排查
request_id	string	该条查询的记录id，主要用于排查问题使用
result	string	实际返回结果，包括查询耗时searchtime、引擎总结果数total、本次请求返回结果数num、本次查询最大返回结果数viewtotal、查询结果items、统计结果facet等信息
errors	string	错误内容，error_message代表错误信息。error_code代表错误码

- searchtime指引擎耗时，单位为秒。
- items包含两个节点fields及variableValue，其中fields为搜索返回字段内容，variableValue为自定义参数返回结果，如获取distance距离值。
- variableValue节点只有在config子句的format为“xml”或者“fulljson”时才能展现出来，“json”格式默认不展示。
- total、viewtotal、num区别：total为一次查询（不考虑config子句）引擎中符合条件的结果数（在结果数较多情况下，该值会做优化），但考虑到性能及相关性，引擎最多会返回viewtotal个结果，如果需要翻页的话，要求start+hit一定要小于viewtotal，total一般用来做展示。num为本次查询请求（受config子句的start及hit）实际返回的条目，不会超过hit值。

示例

请求：（此处省略了公共参数及编码等因素）

```
http://$host/search?index_name=bbs&query=config=start:0,hit:10,format:fulljson&&query=default:'的
'&&filter=create_timestamp>1423000000&&sort=+type;-
RANK&fetch_fields=id;title;body;url;type;create_timestamp&first_formula_name=first_bbs&formula_name=second_
bbs&summary=summary_snipped:1,summary_field:title,summary_element:high,summary_len:32,summary_ellipsis:...;
summary_snipped:2,summary_field:body,summary_element:high,summary_len:60,summary_ellipsis:...
```

成功返回：

```
{
"status": "OK",
"request_id":"1422348642065805107373 58"
"result": {
"searchtime": 0.068196,
"total": 12034,
"num": 2,
"viewtotal": 5000,
"items": [
{
"fields": {
"id": "10",
"type": "10",
"title": "广大中小企业都有各种结构化<high>的</high>数据...",
"body": "广大中小企业都有各种结构化<high>的</high>数据需要进行检索，目前一般采用...",
"url": "http://www.aliyun.com",
"create_timestamp": "21",
"index_name": "bbs"
},
"variableValue": {}
},
{
"fields": {
"id": "14",
"type": "1",
"title": "根据对各种类型典型站点<high>的</high>调研而...",
"body": "根据对各种类型典型站点<high>的</high>调研而制定<high>的</high>一套具有一定扩展性<high>的
</high>结构...",
"url": "http://www.aliyun.com",
"create_timestamp": "25",
"index_name": "bbs"
},
"variableValue": {}
}
],
"facet": []
},
"errors": [],
"tracer": ""
}
```

错误返回：

```
{
"status": "FAIL",
"request_id":"1422348642065805100387587"
"result": {
"searchtime": 0.001488,
"total": 0,
"num": 0,
"viewtotal": 0,
"items": [],
"facet": []
},
"errors": [
{
"code": 6101,
"message": "Index in query clause not exist."
}
],
"tracer": ""
}
```

多应用联合查询

1. 查询：查询语句必须一致（应用结构可以不一致），该查询会往指定的多个应用分别发送，所以必须保证该查询条件（包含first_formula_name、formula_name及summary等参数）在每个应用上都可以使用。一旦个别应用出现问题，则会报错，并返回其他应用结果。
2. 排序：如果用户查询语句中包含sort子句，则获取各个应用结果后按照指定的sort子句再进行一次二次排序；如果没有包含sort子句，则会将各个应用结果交叉展示；如果需要按照相关性（排序表达式）分数进行排序的话，可以显式的在查询语句中指定sort=-RANK。
3. 费用计算：对每个应用分别计算一次访问请求。
4. 最多支持6个应用查询，超过6个则直接报错无结果。

扫描

传统搜索场景的主要目的是为了尽量短的时间内召回最符合的结果，所以对搜索结果进行了限制。在某些场景下需要提供更多的结果来进行分析工作，可以使用scroll接口来获取更多的结果，**目前scorll只支持query与filter子句，sort子句无法支持。**

URL

第一次查询：/search?scroll=1m&search_type=scan

后续查询：/search?scroll=1m&scroll_id=$scroll_id

支持格式

JSON

HTTP请求方式

GET

请求参数

参数	类型	必需	取值范围	默认值	描述
scroll	INT	是			用来表示scroll请求的有效期，默认时间单位为ms, 也可以用1m表示1min；支持的时间单位包括：w=Week, d=Day, h=Hour, m=minute, s=second
search_type	STRING		scan		第一次查询的时候必须填写，后续无需填写
scroll_id	string				第一次不需要，每次搜索结果会返回scroll_id值，后续查询必填

返回结果

参数	类型	描述
status	string	执行结果，OK为成功，FAIL为失败，请根据返回错误码进行排查
request_id	string	该条查询的记录id，主要用于排查问题使用
result	string	实际返回结果，包括查询耗时searchtime、引擎总结果数total、本次请求返回结果数num、本次查询最大返回结果数viewtotal、查询结果items、统计结果facet、scorllid等信息

errors	string	错误内容，error_message代表错误信息。error_code代表错误码

- 返回结果格式目前只支持json。

示例

第一次请求：（此处省略了公共参数及编码等因素）

```
http://$host/search?scroll=1m&search_type=scan&index_name=bbs&query=config=start:0,hit:1,format=fulljson&
&query=default:'的'&&filter=create_timestamp>1423000000
```

成功返回：

```
{
"status":"OK",
"request_id":"142134864206580510037358 7"
"result":{
"searchtime":0.014729,
"total":2,
"num":0,
"viewtotal":2,
"scroll_id":"eJxljsESgjAMRL+mPUNxQA89MHJxPPoBmVgCoqXFtjjw90b05kwO2cnL7tbGUIxnWk+tjoTB3CAadPCcKay
A2xUetEqwGBOYOUQfdCaZChZomSANI+l8p8rykFX7LFO57HwYMel79E4OrqUFHDL07/9CO7SYSG5SG++6oRdF8zU
QRb2MVqhjCmiIVUPXuReq5NkemGypw9l+UKGqyxbAi+R23lqdj/IXmtaJC3DyG9e2Voo=",
"items":[],
"facet":[]
},
"errors":[],
"tracer":""
}
```

后续请求：（此处省略了公共参数及编码等因素）

```
http://$host/search?scroll=1m&scroll_id=eJxljsESgjAMRL+mPUNxQA89MHJxPPoBmVgCoqXFtjjw90b05kwO2cnL7t
bGUIxnWk+tjoTB3CAadPCcKayA2xUetEqwGBOYOUQfdCaZChZomSANI+l8p8rykFX7LFO57HwYMel79E4OrqUFHDL0
7/9CO7SYSG5SG++6oRdF8zUQRb2MVqhjCmiIVUPXuReq5NkemGypw9l+UKGqyxbAi+R23lqdj/IXmtaJC3DyG9e2Vo
o=
```

返回结果：

```
{
"status":"OK",
"request_id":"198734864206580510037358 7"
"result":{
"searchtime":0.010237,
"total":1,
"num":1,
```

```
"viewtotal":1,
"scroll_id":"eJxljbtuwzAMRb9G2gokki25g4agXoqM+QCCkelUjSwnegT230dx0KkAB17wXJ6DtZTSkdbvwSTCaH8gWQxwLxRXwO0KV1o5eEwZbIlpjuZDK9mqpmtFp8WnaMRu1/BajB5ouUF2E5l9I/ReKqVb2Uk+znHCbH7THLgLAy0QsEL/lQ/0bsBMfIvGzmF0Fyb79wMmD8vkmfjKES3V1NO5XJhQdbZCJQcasfgXyoQ+bYK6vBnMObpzybXa3678T2ZG9ImeMdFcYw==",
"items":[
{
"field1":"content1",
"index_name":"app_name"
}
],
"facet":[]
},
"errors":[],
"tracer":""
}
```


注意事项

- start值无效，通过hit值设置每次返回的结果数，即后续查询都以第一次查询指定的hit值为准；
- aggregate、sort、distinct、排序表达式无效，如果传入，查询会报错且无结果；
- 第一次查询需要完整的query、index_name、AccessKeyId等参数，后面的查询不需要传这些参数（即使传入，也会被忽略），只需要传入上一次返回的scroll_id即可；
- 不支持多应用scroll查询；
- 每次查询都必须传scroll参数，如果不传，对于第一次查询，就按正常的查询；对于后续的查询，按scroll处理，但结果中无scroll_id返回。
- 返回结果均有第一次查询中的format决定，后续传scroll_id的响应格式均同第一次；
- 如果用户传入的scroll_id是非法的，那么查询会报错，返回结果格式为json。
- 第一次查询将不返回实际文档数据，只返回scroll_id，需要再次访问才能拿到搜索结果。

搜索

下拉提示是搜索服务的基础功能，在用户输入查询词的过程中，智能推荐候选query，减少用户输入，帮助用户尽快找到想要的内容。 OpenSearch下拉提示在实现了中文前缀，拼音全拼，拼音首字母简拼查询等通用功能的基础上，实现了基于用户文档内容的query智能识别。用户通过控制台的简单配置，就能拥有专属的定制下拉提示。此外，控制台上还提供了黑名单，推荐词条功能，让用户进一步控制下拉提示的结果，实现更灵活的定制。

URL

/suggest

支持格式

JSON

HTTP请求方式

GET

请求参数

参数	类型	必需	取值范围	默认值	描述
query	string	是			需要做下拉提示推荐的query文本，不需要指定索引名。
index_name	string	是			要查询的应用名，不支持多应用查询。
suggest_name	string	是			需要使用的下拉提示规则名称，不支持多个规则同时查询。
hit	int	否	[1-10]	10	需要推荐的query数量

- 下拉提示的query参数和search接口的query参数不同。下拉提示的query参数就是用户在输入框中输入的原始文本，不需要索引字段名，直接query=mp3（mp3为用户输入查询词）即可。

返回结果

参数	类型	描述
request_id	string	该条查询的记录id，主要用于排查问题使用
searchtime	float	引擎查询耗时，单位为秒
suggestions	array	下拉提示结果。array的每个值代表一个结果
errors	string	错误内容，error_message代表错误信息。error_code代表错误码

示例

请求：（此处省略了公共参数及编码等因素）

```
http://$host/suggest?index_name=test_app&suggest_name=test_suggest&hit=3&query=连衣
```

成功返回：

```
{
  "request_id":"1436711330177907890286 24",
  "searchtime":0.002419,
  "suggestions":[
   {
     "suggestion":"连衣裙"
   },
   {
     "suggestion":"连衣裙春装"
   },
   {
     "suggestion":"连衣裙文艺"
   }
  ]
}
```

错误返回：

```
{
  "request_id":"1436709939177907890286 00",
  "searchtime":0.048228,
  "suggestions":[],
  "errors":[
   {
     "code":2551,
     "message":"No such suggestion"
   }
  ]
}
```

索引重建

支持新增、更新、删除的批量操作

URL

/index/$app_name （ app_name为要操作的应用名称 ）

支持格式

JSON

HTTP请求方式

GET

请求参数

参数	类型	必需	取值范围	默认值	描述
action	string	是	createtask		创建索引重建任务
operate	string	否	import		本次索引重建要关联数据导入
table_name	string	否			要导入的表名，多个表之间按照英文逗号(,)分隔

返回结果

示例

只索引重建请求：（此处省略了公共参数及编码等因素）

```
http://$host/index/test?action=createtask
```

成功返回：

索引重建关联数据导入：

```
http://$host/index/test?action=createtask&operate=import&table_name=main
```

成功返回：

配额管理

暂不支持，敬请期待

错误日志

数据处理是异步的，API/SDK推送数据返回OK或者数据源同步的情况下，仍需要及时检查错误日志，数据处理

阶段的错误将在这里展现。

URL

/index/error/$app_name （ app_name为要操作的应用名称 ）

支持格式

JSON

HTTP请求方式

GET、POST

请求参数

参数	类型	必需	取值范围	默认值	描述
page	string	是			目标页码
page_size	int	是			每页返回错误数
sort_mode	string	是	ASC、DESC		指定按照时间升降序排列

返回结果

参数	类型	描述
status	string	执行结果，OK为成功，FAIL为失败，请根据返回错误码进行排查
count	int	错误条数
tems	string	错误内容，cmd代表何种操作，fields代表错误文档内容，error_message代表错误信息。error_code代表错误码

示例

请求：（此处省略了公共参数及编码等因素）

```
http://$host/index/error/test_create_index?page=1&page_size=10&sort_mode=ASC
```

成功返回：

{"result":{"page":1,"page_size":"10","count":0,"items":[]},"status":"OK","RequestId":"142234959609398370021648 7"}

错误返回：

{"result":{},"status":"FAIL","errors":[{"code":2001,"message":"App is not found"}],"RequestId":"142234963402927020084045 5"}

搜索子句介绍

子句说明

config部分可以指定查询结果的起始位置、返回结果的数量、展现结果的格式、参与精排表达式文档个数等。

语法说明

参数	类型	必需	取值范围	默认值	描述
start	int	否	[0, 5000]	0	从搜索结果中第start个文档开始返回
hit	int	否	[0, 500]	10	返回文档的最大数量
format	string	否		json	返回的文档格式，有xml、json、fulljson三种格式可选。fulljson：比json类型多输出一些节点，如variableValue等。
rerank_size	int	否	[0, 2000]	200	设置参与精排个数

注意事项

1. config子句为可选子句；
2. 参数对之间使用逗号(,)分隔；
3. 参数对内key、value采用冒号(:)分隔；
4. start+hit<=5000，超过5000会直接报错无结果。

示例

翻页使用：每页20个结果，分别获取第一页、第二页结果；

```
#第一页
config=start:0, hit:20, format:xml

#第二页
config=start:20, hit:20, format:xml
```

2. 设置精排文档数为1000；

```
config=start:0, hit:20, rerank_size:1000
```

子句说明

query子句是搜索语句中不可缺少的一部分，它表示在哪个索引字段下查询什么内容，并且可以指定多个查询条件及其之间的关系（AND\OR\ANDNOT\RANK）。例如可以将text类型的title和subject组合在一起共同建索引字段default，那么在default上查询时，包含在title或者subject中的关键词都可以找到该文档。有人可以将title字段单独建立索引title_search，那么在title_search上查询时，只有包含在title字段上的关键词才能找到该文档。

语法说明

查询条件格式： 索引字段:’ 关键词’ ^boost。表示在哪个索引字段包含的源字段上查找包含“关键词”的文档。

- 索引字段为定义应用结构索引表时勾选可搜索，后面填写的索引到的字段；表示在哪个字段集合上进行查询；
- 关键词即为要查询的内容。
- boost，为要设置的关键词权重，类型为int，范围为[0,99]，不设置默认为99。
- 查询条件可以为多个，且支持关系为：()、AND、OR、ANDNOT、RANK。(必须为大写)
- 索引字段:”关键词” ，如果使用双引号查询的话为phrase（短语）查询，即要求查询词分词后需要各个term的位置相连、顺序一致。

注意事项

1. query子句为必选子句；
2. 索引字段为定义应用结构索引表时勾选可搜索，后面填写的索引到的字段；
3. 关键词查询必须用单（双）引号括起来，否则会导致报错无结果或者行为不可预期。
4. 只有TEXT、MWS_TEXT、NWS_TEXT、SWS_TEXT类型的字段可以建立组合索引字段；
5. boost值小于0则按照0计算，大于99按照99计算；
6. 关于各种类型的搜索含义，请参见字段类型介绍。

示例

1. 查询title索引字段中包含“北京大学”的文档；

```
query=title:'北京大学'
```

2. 查询title索引字段中包含“北京大学”的文档，并且包含“浙江大学”的文档；

```
query=title:'北京大学' AND title:'浙江大学'
```

3. 查询title索引字段中包含“北京大学”或者“浙江大学” ，且type为“1”的文档；

```
query=(title:'北京大学' OR title:'浙江大学') AND type:'1'
```

4. 查询title索引字段中包含“北京大学”且不包含“清华”的文档，若title中包含“校长”则排序上排在前面；

```
query=(title:'北京大学' ANDNOT title:'清华') RANK title:'校长'
// 精排表达式为：text_relevance(title)
```

5. 查询title索引字段中包含“北京大学”的文档，要求“北京大学”不能分开，不希望返回类似“北京的大学”的文档；

```
query=title:"北京大学"
```

子句说明

过滤功能支持用户根据查询条件，筛选出用户感兴趣的文档。会在通过query子句查找到的文档进行进一步的

过滤，以返回最终所需结果。

语法说明

过滤条件格式为：filed=value

- 过滤条件支持>、<、=、<=、>=、!=、in/notin 等常见条件运算符；以及+、-、*、/、&、^、| 等算术运算符；
- 过滤条件可以配置多个，通过AND、OR及()的逻辑运算关系（必须大写！）进行连接。

注意事项

1. filter为非必选子句；
2. 在filter中出现的字段必须在定义应用结构的时候配置为属性字段；
3. float、double类型因为精度问题无法做精确相等的判断，如有这种场景请，如有这种场景请改用<及>来实现。
4. literal类型的字段值，在filter子句中必须要加双引号（否者会报6135 常量表达式类型错误），支持所有的关系运算，不支持算术运算。
5. 排序特征function函数也可以在filter子句中使用；
6. literal类型字段的过滤仅支持=、!=运算，含义为等于、不等于，不支持>、<等关系运算。（literal字段类型不分词，需要完全匹配）
7. in/notin 判断字段值是否（不）在指定列表中，只支持INT及FLOAT类型，（不支持ARRAY及LITERAL、TEXT、模糊分词系列类型），详细用法参考 搜索相关性函数 中的描述。
8. 分词字段类型无法配置为属性，例如 TEXT，SHORT_TEXT等都不支持，只支持数值字段类型及不分词字段类型配置为属性，例如
   int，int_array，float，float_array，double，double_array，literal，literal_array这些字段类型都支持。

示例

应用中有一个int32字段category，值分别为1(news)，2(bbs)等，需要查询分类category为2(bbs)且包含“浙大”的文档：

```
query=default:'浙大' AND category_search:'2' //category字段创建了一个索引字段category_search
//或者
query=default:'浙大'&&filter=category=2
```

小说应用中有个string_array字段tags，表示小说风格标签：“宫廷”、“悬疑恐怖”、“言情”，需要查询包含“甄嬛传”且标签包含“宫廷”的文档：

```
query=default:'甄嬛传'&&filter=tags="宫廷"
```

电商应用中有个int32的字段hit(点击)和sale(销量)，及int64的字段create_time，需要查询包含“连衣裙”且销量与点击的和乘上rate(偏移率)超过10000，创建时间早于1402345600的文档：

```
query=default:'连衣裙'&&filter=(hit+sale)*rate>10000 AND create_time<1402345600
```

判断用户是否在商家的配送范围。如商家配送范围的字段为coordinates, 用户位置坐标为 (120.307234, 39.294245)，则过滤在配送范围内的商家查询可写为：

```
query=default:'美食'&&filter=in_polygon(coordinates, 120.307234, 39.294245)>0
```

子句说明

用户可以通过查询语句控制结果的排序方式，包括指定排序的字段和升降序。

语法说明

排序子句格式为：+field1;-field2

- field为要排序的字段，+为按字段值升序排序，-为降序排序；
- field也支持简单的算术运算，如+、-、*、\等，但参与运算的字段类型必须一致；
- 支持多维排序，中间用分号（;）分隔；多维排序的含义为，先按照第一维分数排序，如果第一维分数一样，再按照第二维分数进行档内排序，以此类推。
- field部分也可以为“RANK”，表示按照相关性（即排序表达式的计算分值）进行排序。

注意事项

1. sort为非必选子句。如果不填，则默认为sort=-RANK（按照相关性分值降序返回结果）；如果显式使用了sort子句，且子句中不包含RANK，那么定义了排序表达式也不会起作用；
2. 在sort中出现的字段必须在定义应用结构的时候，创建为属性；
3. 返回为int或者float类型的排序特征function函数也可以在sort子句中使用；
4. string类型的字段按照字典序进行排列；
5. 大部分场景下array类型字段均不支持。

示例

查找应用中包含“浙大”的文档，并按照type进行升序排序，如果type相同，则按照文本相关性进行排序：

```
query=default:'浙大'&&sort=+type;-RANK  //精排表达式可以为text_relevance(fileId)
```

查找应用中包含“浙大”的文档，并按照hits（点击）和comments（评论数）总和降序排序:

```
query=default:'浙大'&&sort=-(hits+comments)
```

文档中包含“外婆家”的商家（文档），并且按照用户（120.34256,30.56982）距离商家（lon, lat）的距离进行由近及远的排序:

```
query=default:'外婆家'&&sort=+distance(lon,lat,"120.34256","30.56982")
```

子句说明

一个关键词查询后可能会找到数以万计的文档，用户不太可能浏览所有的文档来获取自己需要的信息，有些情况下用户感兴趣的可能是一些统计的信息。

语法说明

统计子句格式为：group_key:field, range:number1~number2, agg_fun:func1#func2, max_group:number2, agg_filter:filter_clause, max_group:number

- group_key：必选参数。field为要进行统计的字段名，必须配置属性字段，目前支持int类及literal类型的字段做统计。
- agg_fun：必选参数。func可以为count()、sum(id)、max(id)、min(id)四种系统函数，含义分别为：文档个数、对id字段求和、取id字段最大值、取id字段最小值；支持同时进行多个函数的统计，中间用英文井号（#）分隔；sum、max、min的内容支持基本的算术运算；
- range：表示分段统计，可用于分布统计，只支持单个range参数。表示number1~number2及大于number2的区间情况。不支持string类型的字段分布统计。
- agg_filter：非必须参数，表示仅统计满足特定条件的文档；
- agg_sampler_threshold：非必须参数，抽样统计的阈值。表示该值之前的文档会依次统计，该值之后的文档会进行抽样统计；
- agg_sampler_step：非必须参数，抽样统计的步长。表示从agg_sampler_threshold后的文档将间隔agg_sampler_step个文档统计一次。对于sum和count类型的统计会把阈值后的抽样统计结果最后乘以步长进行估算，估算的结果再加上阈值前的统计结果就是最后的统计结果。
- max_group：最大返回组数，默认为1000。

注意事项

1. aggregate为非必选子句；
2. 在aggregate中出现的字段必须在定义应用结构的时候配置为属性字段；
3. aggregate结果会在搜索节点facet节点中展示出来，具体值字段名为agg_fun的名字，如sum、count等
4. aggregate支持多个key的统计，多个统计中间用英文分号（;）分隔。

示例

搜索包含“浙大”的文档，并按照group_id字段进行统计，统计维度包含对price字段进行求和及计算最大值；并对company_id进行统计个数：

```
query=default:'浙大'&&aggregate=group_key:group_id,agg_fun:sum(price)#max(price);group_key:company_id,agg_fun:count()
```

返回结果为：

```
{
status: "OK",
result: {
searchtime: 0.015634,
total: 5,
num: 1,
viewtotal: 5,
items: [ //具体搜索结果
{ ... }
],
facet: [
{
key: "group_id",
items: [
{
value: 43,
sum: 81,
max: 20,
},
{
value: 63,
sum: 91,
max: 50,
},
],
},
{
key: "company_id",
items: [
{
value: 13,
count: 4,
},
{
value: 10,
count: 1,
},
],
},
],
},
errors: [ ],
tracer: "",
```

```
},
```

搜索包含“浙大”的文档，并按照group_id字段进行统计，统计维度包含对price字段进行求和。其中10000以后的文档进行抽样，步长为5：

```
query=default:'浙大'&&aggregate=group_key:group_id,agg_fun:sum(price), agg_sampler_threshold:10000, agg_sampler_step:5
```

搜索包含“浙大”的文档，并按照group_id字段进行统计个数，统计维度为小于10、10~50、及大于50的文档数；

```
query=default:'浙大'&&aggregate=group_key:group_id,agg_fun:count(),range:10~50
```

搜索包含“浙大”的文档，并按照group_id字段进行统计hits及replies的和的最大值，仅统计create_timestamp大于1423456781的文档；

```
query=default:'浙大'&&aggregate=group_key:group_id,agg_fun:max(hits+replies),agg_filter:create_timestamp>1423456781
```

子句说明

聚合子句可以在一定程度上保证展示结果的多样性，以提升用户体验。如一次查询可以查出很多的文档，但是如果某个用户的多个文档分值都比较高，则都排在了前面，导致一页中所展示的结果几乎都属于同一用户，这样既不利于结果展示也不利于用户体验。对此，聚合子句可以对每个用户的文档进行抽取，使得每个用户都有展示文档的机会。

语法说明

子句语法格式为：dist_key:field,dist_count:1,dist_times:1,reserved:false

参数	类型	必需	取值范围	默认值	描述
dist_key	string	是			要聚合的字段
dist_times	int	否		1	抽取的轮数
dist_count	int	否		1	一轮抽取的文档数
reserved	true/false	否	true/false	true	是否保留抽取之后剩余的文档。如果为false，为不保留，则搜索结果的

					total（总匹配结果数）会不准确。
update_total_hit	true/false	否	true/false	false	当reserved为false时，设置update_total_hit为true，则最终total_hit会减去被distinct丢弃的的数目（不一定准确），为false则不减。
dist_filter	string	否			过滤条件，被过滤的doc不参与distinct，只在后面的排序中，这些被过滤的doc将和被distinct出来的第一组doc一起参与排序。默认是全部参与distinct。
grade	float	否			指定档位划分阈值，所有的文档将根据档位划分阈值划分成若干档，每个档位中各自根据distinct参数做distinct，可以不指定该参数，默认是所有文档都在同一档。档位的划分按照文档排序时第一维的排序依据的分数进行划分，两个档位阈值之间用“|”

					分开，档位的个数没有限制。例如：1、grade:3.0：表示根据第一维排序依据的分数分成两档，(< 3.0)的是第一档，(>= 3.0)的是第二档；2、grade:3.0|5.0 ：表示分成三档，(< 3.0)是第一档，(>= 3.0，< 5.0)是第二档，(>= 5.0)是第三档。档位的先后顺序和第一维排序依据的顺序一致，即如果第一维排序依据是降序，则档位也是降序，反之亦然。

注意事项

1. distinct为非必选子句；
2. 在distinct中出现的字段必须在定义应用结构的时配置为属性字段；；
3. 不支持array类型，只支持int和literal字段类型。

distinct uniq插件

如上面描述，如果reserved=false情况下，会导致搜索结果中的total及viewtotal不准确，如果用户需要依赖于这个值进行翻页或者其他处理，则会有问题。为此，系统提供了distinct uniq的插件来解决在dist_times:1,dist_count:1,reserved:false的情况下的total及viewtotal展示不准确。 在kvpairs中添加duniqfield:field即可。

注意：

- field必须与distinct子句中的dist_key一致；
- 该插件仅在在dist_times:1,dist_count:1,reserved:false查询下起作用，任何参数值有变化都将无效。

- 出于性能考虑，目前该插件最大支持total值为5000，即使真实搜索结果数超过5000，也会返回5000。

示例

查看create_time（创建时间）在1402301230之后且包含“浙大”的文档，并按照company_id字段进行聚合抽取10轮，每轮取2个结果，抽取后的文档排在后面：

```
query=default:'浙大'&&filter=create_time>1402301230&&distinct=dist_key:company_id,dist_count:2,dist_times:10
```

查看包含“浙大”的文档，且按照company_id字段进行聚合抽取1轮，每轮取1个结果，抽取后的文档丢弃（要求返回结果数为丢弃后的）：

```
query=default:'浙大'&&distinct=dist_key:company_id,dist_count:1,dist_times:1,reserved:false&&kvpairs=duniqfield:company_id
```

自定义参数kvpairs子句

子句说明

为便于通过查询语句传递信息给具体的特征函数，用户可以在kvpairs子句中对排序表达式中的可变部分进行参数定义。

语法说明

kvpairs子句格式为：key1:value1,key2:value2

- key1为kvpairs中的字段，value1为key1字段对应的值，二者之间使用英文冒号（:）分隔；
- 根据需要，可以定义多个key-value对，使用英文逗号（,）分隔；

注意事项

1. kvpairs子句为可选子句；

示例

查看包含“浙大”的文档，且按照company_id字段进行聚合抽取1轮，每轮取1个结果，抽取后的文档丢弃

（要求返回结果数为丢弃后的）：

```
query=default:'浙大
'&&distinct=dist_key:company_id,dist_count:1,dist_times:1,reserved:false&&kvpairs=duniqfield:company_id
```

查看包含“浙大”的文档，且对其他tags字段（int_array类型，包含若干属性对）进行key1=13、value1=10、key2=100、value2=1的相关性特征函数—tag_match匹配，如果匹配，则计算sum，多个key命中取最大值：

```
query=default:'浙大'&&kvpairs=query_key:13=10:100=1  //精排表达式为tag_match(query_key, tags, sum, max, true, true)
```