Open Table Service RESTful API

目录

API 约束 2
- 用户身份认证 2
- Request 签名 2
- Response 签名 4
- Request 参数传递 5
- Request 的返回方式 7

API 及参数描述 8
- 表相关操作 9
- 事务相关操作 15
- 查询操作 17
- 更改操作 25

错误信息 30
- 一般错误 31
  - 身份认证错误 31
  - 一般请求错误 31
  - 计量相关错误 32
- API 特定错误 32

API 约束

用户身份认证

OTS通过对称签名的方法来验证某个请求是其拥有者发送，以及应答是OTS所发送。当用户在OTS注册之后，OTS会提供一对AccessID和AccessKey。AccessID用来标识用户，AccessKey是用来对请求和响应进行签名和验证的密钥。AccessKey需保密，只有OTS和用户知道。用户以个人身份向OTS发送请求时，需要包括请求明文，AccessID和使用AccessKey对请求明文中的信息部分签名产生的验证码。OTS收到请求后，会通过AccessID找到相应的AccessKey，以同样的方式签名明文中的信息。如果计算出来的验证码和提供的验证码相同，则认为是有效的用户发送的请求。验证OTS的应答时需要用户使用相同的方式进行计算，若计算的验证码和提供的验证码一致，则用户可以认为应答是有效的OTS应答。

Request 签名

OTS的请求和应答是使用HTTP协议进行传递的。Request签名仅支持URI(Uniform Resource Identifier)签名一种方式，用户在任何一个Request中必须包含6个参数：APIVersion，Date，OTSAccessKeyId，SignatureMethod，SignatureVersion，Signature。

APIVersion表示目前OTS API的版本，其值只能是1。

Date参数有3种可选的形式：

1. %a, %d %b %Y %H:%M:%S GMT (如：Mon, 3 Jan 2010 08:33:47 GMT)
2. %A, %d-%b-%y %H:%M:%S GMT (如：Monday, 3-Jan-10 08:33:47 GMT)
3. %a %b %d %H:%M:%S %Y(如：Mon Jan 3 08:33:47 2010)

这些时间传递的都是UTC时间。Date所表示的时间与服务器接收到request的时间最大误差为15分钟。如果超过15分钟的时间误差则服务器端报错：OTSAuthFailed。

OTSAccessKeyId就是OTS提供的AccessID。

SignatureMethod表示计算签名的算法，目前只支持值为HmacSHA1这一种。

SignatureVersion表示计算签名算法的版本，目前只支持值为1的版本。

Signature即是对整个URL进行签名后的字符串。用于签名的字符串包括用户请求的资源名以及请求的参数。参数的名称和值经过URL encoding后，之间用’=’（等号）相隔，组成字符串并对参数名-值对按照字典序排序后，以’&’符号连接构成字符串。签名的计算方法如下：

```
Base64(hmac-sha1(VERB + “\n” + “KEY1=VALUE1” + “&” + “KEY2=VALUE2”, AccessKey))
```

其中VERB表示请求的资源名，之后接参数的名称和值。

例如，用户的请求为：

```
http://service.ots.aliyun.com/CreateTable?
TableName=CapTable
&PK.1.Name=PrimaryKey1
&PK.1.Type=STRING
&PK.2.Name=PrimaryKey2
&PK.2.Type=INTEGER
&View.1.Name=View1
&View.1.PK.1.Name=PrimaryKey1
&View.1.PK.1.Type=STRING
&View.1.PK.2.Name=Column1
&View.1.PK.2.Type=BOOLEAN
&View.1.Column.1.Name=Column2
&View.1.Column.1.Type=STRING
&View.1.Column.2.Name=Column3
&View.1.Column.2.Type=DOUBLE
&APIVersion=1
&Date=Fri%2C%2016%20Sep%202006%2018%3A07%3A20%20GMT
&OTSAccessKeyId=ID1
&SignatureMethod=HmacSHA1
&SignatureVersion=1
```

那么用于签名的字符串为：

```
“/CreateTable” + “\n” +
“APIVersion=1” +
“Date=Fri%2C%2016%20Sep%202006%2018%3A07%3A20%20GMT” +
“&OTSAccessKeyId=ID1” +
“&PK.1.Name=PrimaryKey1” +
“&PK.1.Type=STRING” +
“&PK.2.Name=PrimaryKey2” +
“&PK.2.Type=INTEGER” +
“&SignatureMethod=HmacSHA1” +
“&SignatureVersion=1” +
“&TableName=CapTable” +
“&View.1.Column.1.Name=Column2” +
“&View.1.Column.1.Type=STRING” +
“&View.1.Column.2.Name=Column3” +
“&View.1.Column.2.Type=DOUBLE” +
“&View.1.Name=View1” +
```

```
“&View.1.PK.1.Name=PrimaryKey1” +
“&View.1.PK.1.Type=STRING” +
“&View.1.PK.2.Name=Column1” +
“&View.1.PK.2.Type=BOOLEAN”
```

最终传入的形式是（假设ID1对应的AccessKey为：KEY1，下面的签名字符串是用KEY1计算出来的）：

```
http://service.ots.aliyun.com/CreateTable?
TableName=CapTable
&PK.1.Name=PrimaryKey1
&PK.1.Type=STRING
&PK.2.Name=PrimaryKey2
&PK.2.Type=INTEGER
&View.1.Name=View1
&View.1.PK.1.Name=PrimaryKey1
&View.1.PK.1.Type=STRING
&View.1.PK.2.Name=Column1
&View.1.PK.2.Type=BOOLEAN
&View.1.Column.1.Name=Column2
&View.1.Column.1.Type=STRING
&View.1.Column.2.Name=Column3
&View.1.Column.2.Type=DOUBLE
&APIVersion=1
&Date=Fri%2C%2016%20Sep%202006%2018%3A07%3A20%20GMT
&OTSAccessKeyId=ID1
&SignatureMethod=HmacSHA1
&SignatureVersion=1
&Signature=%2FL034xFZBPO%2BNpxA%2BSufMiOt%2BKQ%3D
```

返回结果：

- 如果用户AccessId不存在，则返回403 Forbidden
- 如果用户的签名校验失败，则返回403 Forbidden
- 如果没有传入必要的参数，则返回400 Bad Request
- 传入请求的时间必须在OTS服务器当前时间前后的15分钟内，否则返回403 Forbidden

Response 签名

Response的签名包含在HTTP的header中的Authorization段，表明这个响应被授权。这个Header的样例如下：

```
Authorization:OTS44CF9590006BF252F707:jZNOcbfWmD/A/f3hSvVzXZjM2HU=
```

在该头中用冒号隔开的两个值分别为用户的AccessID和签名。验证码计算方法如下：

```
“Authorization: OTS “ + AccessID + “:” + base64(hmac-sha1(
                        + CONTENT-MD5 + “\n”
                        + CONTENT-TYPE + “\n”
                        + CanonicalizedOTSHeaders
                        + CanonicalizedResource
                       , AccessKey))
```

如上所示，OTS响应的签名的Header包括Content-Md5， Content-Type， 规范化OTS Header(CanonicalizedOTSHeaders)， 规范化OTS资源地址(CanonicalizedResource)。其中必选的Header为Content-Type和Content-MD5，每个Header之后都需要加上回车符，如果没有规范化OTSHeader，则忽略。规范化OTS头为所有以"x-ots-"为前缀的头，在对规范化OTSHeader签名时需要遵循以下规则：

1. Header名字小写
2. Header按名字的字母序从小到大顺序排列
3. 分割Header名和值的冒号前后不能有空格
4. 每个Header之后都有换行(‘\n’)
5. 如果没有规范化OTS Header，则为空。
6. 目前版本中有唯一的规范化OTS头，为x-ots-date。表示OTS给出Response的时间。时间格式为：%a, %d %b %Y %H:%M:%S GMT (如：Mon, 3 Jan 2010 08:33:47 GMT)

用户可以选择使用相同的方式计算签名，并通过比较计算出的Signature和OTS提供的Signature是否一致来确定响应是否有效。

Request 参数传递

OTS 允许Request以HTTP GET和HTTP POST两种方法进行参数传递。

- **GET方法介绍**

GET方法允许参数全部通过URI进行传递。允许直接通过超链接、浏览器地址栏等方式直接进行访问。如上述例子中的地址为：

```
http://service.ots.aliyun.com/CreateTable?TableName=CapTable&PK.1.Name=PrimaryKey1&PK.1.Type=STRING&PK.2.Name=PrimaryKey2&PK.2.Type=INTEGER&View.1.Name=View1&View.1.PK.1.Name=PrimaryKey1&View.1.PK.1.Type=STRING&View.1.PK.2.Name=Column1&View.1.PK.2.Type=BOOLEAN&View.1.Column.1.Name=Column2&View.1.Column.1.Type=STRING&View.1.Column.2.Name=Column3&View.1.Column.2.Type=DOUBLE&OTSAccessKeyId=ID1&APIVersion=1&Date=Fri%2C%2016%20Sep%202006%2018%3A07%3A20%20GMT&SignatureMethod=HmacSHA1&SignatureVersion=1&Signature=%2FL034xFZBPO%2BNpxA%2BSufMiOt%2BKQ%3D
```

通过GET方法访问OTS的地址的URI的长度建议不大于512个字节，否则为了保证URI传递到OTS的正确性，请使用POST方法，POST方法支持的最大请求大小是2M字节。

- **POST方法介绍**

POST方法的URI为GET方法的URI中除了参数名称和参数值之外的其它部分。参数部分需要放在POST的内容里面。例如上述例子中的请求地址是：

```
http://service.ots.aliyun.com/CreateTable
```

POST的参数部分是：

```
TableName=CapTable&PK.1.Name=PrimaryKey1&PK.1.Type=STRING&PK.2.Name=PrimaryKey2&PK.2.Type=INTEGER&View.1.Name=View1&View.1.PK.1.Name=PrimaryKey1&View.1.PK.1.Type=STRING&View.1.PK.2.Name=Column1&View.1.PK.2.Type=BOOLEAN&View.1.Column.1.Name=Column2&View.1.Column.1.Type=STRING&View.1.Column.2.Name=Column3&View.1.Column.2.Type=DOUBLE&OTSAccessKeyId=ID1&APIVersion=1&Date=Fri%2C%2016%20Sep%202006%2018%3A07%3A20%20GMT&SignatureMethod=HmacSHA1&SignatureVersion=1&Signature=%2FL034xFZBPO%2BNpxA%2BSufMiOt%2BKQ%3D
```

OTS 规定所有RESTful请求传递参数的顺序必须严格按照API及参数描述中列出的顺序给出，然后再依次跟上APIVersion，Date，OTSAccessKeyId， SignatureMethod，SignatureVersion， Signature这6个参数。如下的请求就是符合本文档所描述的参数顺序的请求：

```
http://service.ots.aliyun.com/CreateTable?
TableName=CapTable
&PK.1.Name=PrimaryKey1
&PK.1.Type=STRING
&PK.2.Name=PrimaryKey2
&PK.2.Type=INTEGER
&View.1.Name=View1
&View.1.PK.1.Name=PrimaryKey1
&View.1.PK.1.Type=STRING
&View.1.PK.2.Name=Column1
&View.1.PK.2.Type=BOOLEAN
&View.1.Column.1.Name=Column2
&View.1.Column.1.Type=STRING
&View.1.Column.2.Name=Column3
&View.1.Column.2.Type=DOUBLE
&APIVersion=1
&Date=Fri%2C%2016%20Sep%202006%2018%3A07%3A20%20GMT
&OTSAccessKeyId=ID1
&SignatureMethod=HmacSHA1
&SignatureVersion=1
&Signature=%2FL034xFZBPO%2BNpxA%2BSufMiOt%2BKQ%3D
```

如果任意改变参数的顺序，如对View和PK的先后传递顺序进行调换（如下），则此请求会失败。

```
http://service.ots.aliyun.com/CreateTable?
TableName=CapTable
&View.1.Name=View1
&View.1.PK.1.Name=PrimaryKey1
&View.1.PK.1.Type=STRING
&View.1.PK.2.Name=Column1
&View.1.PK.2.Type=BOOLEAN
&View.1.Column.1.Name=Column2
&View.1.Column.1.Type=STRING
&View.1.Column.2.Name=Column3
&View.1.Column.2.Type=DOUBLE
&PK.1.Name=PrimaryKey1
&PK.1.Type=STRING
&PK.2.Name=PrimaryKey2
&PK.2.Type=INTEGER
&APIVersion=1
&Date=Fri%2C%2016%20Sep%202006%2018%3A07%3A20%20GMT
&OTSAccessKeyId=ID1
&SignatureMethod=HmacSHA1
&SignatureVersion=1
&Signature=%2FL034xFZBPO%2BNpxA%2BSufMiOt%2BKQ%3D
```

在使用HTTP GET方法或是POST方法传递参数时需要对URI进行编码，方式见RFC2369。请注意OTS不允许把空格编码为'+'，而必须是'%20'。请注意：字符'&'用作参数的分割符时不需编码，否则须编码为'%26'。

Request 的返回方式

OTS 的所有请求都以UTF-8编码的XML 1.0标准格式返回。API请求成功的返回示例请见本文档下面的部分。失败的错误信息请见 Error Message 小节。

若返回的数据中出现了符合UTF-8但不符合XML 1.0标准的字符，OTS会对相应的Value进行Base64编码，再返回给用户，并标明此Value是经过Base64编码过的。如：

```
<?xml version="1.0" encoding="UTF-8"?>
<GetRowResult>
  <Table name="Table1">
```

```
    <Row>
      <Column PK=”true”>
        <Name>PK1</Name>
        <Value type=”STRING”>a</Value>
      </Column>
      <Column PK=”true”>
        <Name>PK2</Name>
        <Value type=”STRING”>b</Value>
      </Column>
      <Column>
        <Name>C1</Name>
        <Value type=”STRING” encoding=”Base64”>FWEFWEJ=</Value>
      </Column>
    </Row>
  </Table>
  <RequestID>xxxxxxxxxxxxx</RequestID>
  <HostID>xxxxxxxxxxxx</HostID>
</GetRowResult>
```

OTS会标注返回XML中相应的Value段的值的类型：INTEGER， STRING， BOOLEAN，DOUBLE 。 类 型 的 标 注 方 法 分 别 是 ： type=”INTEGER” ， type=”STRING” ，type=”BOOLEAN”， type=”DOUBLE”。如果查询返回的结果中的某列为主键，此列会被标出：<Column PK=”true” />。结果中的RequestID和HostID是排查错误的辅助信息。如果使用OTS碰到问题无法解决，需要联系客服的话，请提供这两个ID。

API 及参数描述

下述小节中出现的参数描述中如出现斜体的 *X*, *Y* 字符，则表示可以传递一系列的此项参数。*X*,*Y* 表示序号，从 1 开始连续递增。序号相同的表示一组参数，一组参数写完后才可以接下一个或下一组参数。如下述的 CreateTableAPI 中的参数 PK.X.Name, PK.X.Type 和 PagingKeyLen。那么合法的顺序为：

```
PK.1.Name=PrimaryKey1
PK.1.Type=STRING
PK.2.Name=PrimaryKey2
PK.2.Type=INTEGER
PagingKeyLen=0
```

下面的例子则是一种非法的顺序:

```
PK.1.Name=PrimaryKey1
PK.2.Name=PrimaryKey2
PK.1.Type=STRING
PK.2.Type=INTEGER
```

```
PagingKeyLen=0
```

表相关操作

- CreateTableGroup
  - 创建表组。
  - 参数
    - TableGroupName
      - 值为表组名。
      - 类型为名称。
      - 必选。
    - PartitionKeyType
      - 表示表组的数据分片键的类型。
      - 类型为枚举，只能为 INTEGER 或者 STRING。
      - 必选。
  - 请求样例

```
http://service.ots.aliyun.com/CreateTableGroup?
TableGroupName=DemoTableGroup
&PartitionKeyType=STRING
&APIVersion=1
&Date=Tue%2C%2027%20Dec%202011%2013%3A36%3A01%20GMT
&OTSAccessKeyId=ID1
&SignatureMethod=HmacSHA1
&SignatureVersion=1
&Signature=XXXXXXXXXX
```

  - 返回成功样例

```
<?xmlversion="1.0" encoding=”UTF-8”?>
<CreateTableGroupResult>
<Code>OK</Code>
<RequestID>xxxxxxxxxxxx</RequestID>
<HostID>xxxxxxxxxxx</HostID>
</CreateTableGroupResult>
```

- ListTableGroup
  - 获取表组的列表。
  - 无参数
  - 请求样例

```
http://service.ots.aliyun.com/ListTableGroup?
APIVersion=1
&Date=Tue%2C%2027%20Dec%202011%2014%3A06%3A03%20GMT
&OTSAccessKeyId=ID1
&SignatureMethod=HmacSHA1
```

```
&SignatureVersion=1
&Signature=XXXXXXXXXXX
```

- 成功返回样例

```
<?xmlversion="1.0" encoding=”UTF-8”?>
<ListTableGroupResult>
<TableGroupNames>
    <TableGroupName>DemoTableGroup1</TableGroupName>
    <TableGroupName>DemoTableGroup2</TableGroupName>
</TableGroupNames>
<RequestID>xxxxxxxxxxxxx</RequestID>
<HostID>xxxxxxxxxxxx</HostID>
</ListTableGroupResult>
```

- 注意:
  - 如果表组名称的列表为空，节点<TableGroupNames />仍然存在。

- DeleteTableGroup
  - 删除表组及属于该表组的相关表和视图。
  - 参数
    - TableGroupName
      - 值为表组名。
      - 类型为名称。
      - 必选。
  - 请求样例

```
http://service.ots.aliyun.com/DeleteTableGroup?
TableGroupName=DemoTableGroup
&APIVersion=1
&Date=Tue%2C%2027%20Dec%202011%2014%3A11%3A04%20GMT
&OTSAccessKeyId=ID1
&SignatureMethod=HmacSHA1
&SignatureVersion=1
&Signature=XXXXXXXXXXX
```

- 返回成功样例

```
<?xmlversion="1.0" encoding=”UTF-8”?>
<DeleteTableGroupResult>
<Code>OK</Code>
<RequestID>xxxxxxxxxxxxx</RequestID>
<HostID>xxxxxxxxxxxx</HostID>
</DeleteTableGroupResult>
```

- CreateTable
  - 创建表并创建相关的视图。
  - 参数
    - TableName

- 值为表名。
- 类型为名称。
- 必选。

◆ PK.*X*.Name
- 值为 PK 列名，*X* 为序号，第 1 个 PK 列为数据分片键。PK 列的名字不能重复。
- 类型为名称。
- 必选，至少 1 个，保证顺序。

◆ PK.*X*.Type
- 值为第 *X* 个 PK 列的数据类型。
- 类型为枚举，只能为 INTEGER， STRING， BOOLEAN 之一。第一个 PK 列的类型不能为 BOOLEAN。如果此表出现在表组中，则第一个 PK 列的类型必须和表组的数据分片键的类型一致。
- 必选，紧跟 PK.*X*.Name 并和 PK.*X*.Name 一一对应。

◆ PagingKeyLen
- 表示是否要建立分页，且分页建立在前几个 PK 列上。0 表示不建立分页键。此参数的值必须小于 PK 列的个数。
- 类型为 INTEGER。
- 可选，默认值为 0。

◆ View.*Y*.Name
- 值为第 *Y* 个视图的名称。
- 类型为名称。
- 可选，不选则表示不建立视图。
- 注意：View.*Y*.Name, View.*Y*.PK.*X*.Name, View.*Y*.PK.*X*.Type, View.*Y*.Column.*X*.Name, View.*Y*.Column.*X*.Type, View.*Y*.PagingKeyLen 为视图相关参数，必须完整写完一个视图的所有参数才能写下一个视图的参数。

◆ View.*Y*.PK.*X*.Name
- 值为要建立的第 *Y* 个视图的第 *X* 个 PK 列的名字。本视图中的列和视图的 PK 列的名字不能重复。
- 类型为名称。
- 若视图 *Y* 出现，则必选，至少 1 个。第 1 个 PK 列的名字必须和该视图所属原表的第 1 个 PK 列的名字一致。
- 注意：必须完整写完一个视图的所有参数才能写下一个视图的参数。

◆ View.*Y*.PK.*X*.Type
- 值为要建立的第 *Y* 个视图的第 *X* 个 PK 列的类型。
- 类型为枚举，只能为 INTEGER， STRING， BOOLEAN 之一。
- 和 View.*Y*.PK.*X*.Name 一一对应。

◆ View.*Y*.Column.*X*.Name
- 值为要建立的第 *Y* 个视图的第 *X* 个属性列的名字。本视图中的列和其 PK 列的名字不能重复。
- 类型为名称。
- 可选，可重复。

- View.*Y*.Column.*X*.Type
  - 值为要建立的第 *Y* 个视图的第 *X* 个属性列的类型。
  - 类型为枚举，只能为 INTEGER， STRING， BOOLEAN， DOUBLE 之一。
  - 可选，和 View.*Y*.Column.*X*.Name 一一对应。
- View.*Y*.PagingKeyLen
  - 表示是否要在当前视图上建立分页，且分页建立在视图的前几个 PK 列上。值为 0 表示不建立分页。
  - 类型为 INTEGER。
  - 可选，默认值为 0。
- TableGroupName
  - 表示此表属于哪个表组。
  - 类型为名称。
  - 可选，若指定，则此表属于指定的表组，否则此表不属于任何表组。

■ 请求样例

```
http://service.ots.aliyun.com/CreateTable?
TableName=Table1
&PK.1.Name=PK1
&PK.1.Type=STRING
&PK.2.Name=PK2
&PK.2.Type=INTEGER
&PagingKeyLen=1
&View.1.Name=View1
&View.1.PK.1.Name=PK1
&View.1.PK.1.Type=STRING
&View.1.PK.2.Name=C1
&View.1.PK.2.Type=STRING
&View.1.PK.3.Name=PK2
&View.1.PK.3.Type=INTEGER
&View.1.Column.1.Name=C2
&View.1.Column.1.Type=STRING
&View.1.PagingKeyLen=0
&TableGroupName=DemoTableGroup
&APIVersion=1
&Date=Tue%2C%2027%20De%%202011%2011%3A20%3A42%20GMT
&OTSAccessKeyId=ID1
&SignatureMethod=HmacSHA1
&SignatureVersion=1
&Signature=XXXXXXXXXXX
```

■ 成功返回样例

```
<?xmlversion="1.0" encoding=”UTF-8”?>
<CreateTableResult>
```

```
  <Code>OK</Code>
  <RequestID>xxxxxxxxxxxxx</RequestID>
  <HostID>xxxxxxxxxxxx</HostID>
</CreateTableResult>
```

- 注意
  - OTS 要求视图中数据行与原表数据行一一对应，即用户必须保证视图中每一行数据的 PK 列的值的唯一性，因此我们强烈建议视图 PK 列包含表的全部 PK 列。

- GetTableMeta
  - 获取表的结构。
  - 参数
    - TableName
      - 值为表名。
      - 类型为名称。
      - 必选。
  - 请求样例

```
http://service.ots.aliyun.com/GetTableMeta?
TableName=Table1
&APIVersion=1
&Date=Tue%2C%2027%20Dec%202011%2011%3A20%3A43%20GMT
&OTSAccessKeyId=ID1
&SignatureMethod=HmacSHA1
&SignatureVersion=1
&Signature=XXXXXXXXXXX
```

  - 成功返回样例

```
<?xml version=”1.0”encoding=”UTF-8”?>
<GetTableMetaResult>
  <TableMeta>
    <TableName>Table1</TableName>
    <PrimaryKey>
      <Name>PK1</Name>
      <Type>STRING</Type>
    </PrimaryKey>
    <PrimaryKey>
      <Name>PK2</Name>
      <Type>INTEGER</Type>
    </PrimaryKey>
    <PagingKeyLen>1</PagingKeyLen>
    <View>
      <Name>View1</Name>
      <PrimaryKey>
        <Name>PK1</Name>
```

```
            <Type>STRING</Type>
        </PrimaryKey>
        <PrimaryKey>
            <Name>C1</Name>
            <Type>STRING</Type>
        </PrimaryKey>
        <PrimaryKey>
            <Name>PK2</Name>
            <Type>INTEGER</Type>
        </PrimaryKey>
        <Column>
            <Name>C2</Name>
            <Type>STRING</Type>
        </Column>
        <PagingKeyLen>0</PagingKeyLen>
    </View>
    <TableGroupName>DemoTableGroup</TableGroupName>
  </TableMeta>
  <RequestID>xxxxxxxxxxxxx</RequestID>
  <HostID>xxxxxxxxxxxx</HostID>
</GetTableMetaResult>
```

- 注意
  - 若此表不在表组中，则<TableMeta/>节点中的<TableGroupName /> 节点会不出现。

- ListTable
  - 列出表名。
  - 无参数
  - 请求样例

```
http://service.ots.aliyun.com/ListTable?
APIVersion=1
&Date=Tue%2C%2027%20Dec%202011%2014%3A11%3A04%20GMT
&OTSAccessKeyId=ID1
&SignatureMethod=HmacSHA1
&SignatureVersion=1
&Signature=XXXXXXXXXXX
```

  - 成功返回样例

```
<?xmlversion="1.0" encoding=”UTF-8”?>
<ListTableResult>
  <TableNames>
    <TableName>Table1</TableName>
    <TableName>Table2</TableName>
    <TableName>Table3</TableName>
```

```
  </TableNames>
  <RequestID>xxxxxxxxxxxxx</RequestID>
  <HostID>xxxxxxxxxxxx</HostID>
</ListTableResult>
```

- 注意:
  - 如果表名称的列表为空，节点<TableNames />仍然存在。

- DeleteTable
  - 删除表及和此表一起创建的视图。
  - 参数
    - TableName
      - 值为表名。
      - 类型为名称。
      - 必选。
  - 请求样例

```
http://service.ots.aliyun.com/DeleteTable?
TableName=Table1
&APIVersion=1
&Date=Tue%2C%2027%20Dec%202011%2011%3A20%3A42%20GMT
&OTSAccessKeyId=ID1
&SignatureMethod=HmacSHA1
&SignatureVersion=1
&Signature=XXXXXXXXXXX
```

  - 成功返回样例

```
<?xmlversion="1.0" encoding=”UTF-8”?>
<DeleteTableResult>
  <Code>OK</Code>
  <RequestID>xxxxxxxxxxxxx</RequestID>
  <HostID>xxxxxxxxxxxx</HostID>
</DeleteTableResult>
```

事务相关操作

- StartTransaction
  - 在表或表组上开始一个事务，并创建事务 ID。用户必须指定数据分片键，并保证所有在这个事务中的操作的数据分片键的值等于在此 API 中指定的数据分片键的值。若此数据分片键已经存在另一个事务中且该事务没有完成或被取消，则本次事务会直接失败，用户可以重试。
  - 参数
    - EntityName
      - 值为表名或表组名。
      - 类型为名称。
      - 必选。

- PartitionKeyValue
  - 表示事务建立在哪个数据分片键的值之上。
  - 类型由参数 PartitionKeyType 指定。
  - 必选。
- PartitionKeyType
  - 表示 PartitionKeyValue 的类型。
  - 类型为枚举，只能为 INTEGER， STRING 之一，与 EntityName 指定的表或表组的数据分片键类型一致。
  - 必选。

■ 请求样例

```
http://service.ots.aliyun.com/StartTransaction?
EntityName=Table1
&PartitionKeyValue=%27a%27
&PartitionKeyType=STRING
&APIVersion=1
&Date=Tue%2C%2027%20Dec%202011%2011%3A25%3A45%20GMT
&OTSAccessKeyId=ID1
&SignatureMethod=HmacSHA1
&SignatureVersion=1
&Signature=XXXXXXXXXXX
```

■ 成功返回样例：

```
<?xml version="1.0" encoding=”UTF-8”?>
<StartTransactionResult>

<TransactionID>DQAAAGNvbnRhY3RfdGFibGUbAAAAMTE2NjM5NTcxMi0xMzIyNDY3
OTI5NTQ4MDQ2AwQAAAB1aWQxdzznkA==</TransactionID>
  <RequestID>xxxxxxxxxxxxx</RequestID>
  <HostID>xxxxxxxxxxxx</HostID>
</StartTransactionResult>
```

- CommitTransaction
  - ■ 确认并提交事务，提交后此事务 ID 失效。
  - ■ 参数
    - TransactionID
      - 值为 StartTransactionAPI 给出的事务 ID。
      - 类型为合法的事务 ID。
      - 必选。
  - ■ 请求样例

```
http://service.ots.aliyun.com/CommitTransaction?
TransactionID=DQAAAGNvbnRhY3RfdGFibGUbAAAAMTE2NjM5NTcxMi0xMzIyNDY3O
TI5NTQ4MDQ2AwQAAAB1aWQxdzznkA==
&APIVersion=1
&Date=Tue%2C%2027%20Dec%202011%2011%3A25%3A45%20GMT
```

```
&OTSAccessKeyId=ID1
&SignatureMethod=HmacSHA1
&SignatureVersion=1
&Signature=XXXXXXXXXXX
```

- 成功返回样例：

```
<?xml version="1.0" encoding=”UTF-8”?>
<CommitTransactionResult>
  <Code>OK</Code>
  <RequestID>xxxxxxxxxxxxx</RequestID>
  <HostID>xxxxxxxxxxxx</HostID>
</CommitTransactionResult>
```

- AbortTransaction
  - 撤销一个事务，撤销后所有在此事务中的操作的结果都被取消。撤销后此事务 ID 失效。
  - 参数
    - TransactionID
      - 值为 StartTransactionAPI 给出的事务 ID。
      - 类型为合法的事务 ID。
      - 必选。
  - 请求样例

```
http://service.ots.aliyun.com/AbortTransaction?
TransactionID=DQAAAGNvbnRhY3RfdGFibGUbAAAAMTE2NjM5NTcxMi0xMzIyNDY3OTI5NTQ4MDQ2AwQAAAB1aWQxdzznkA==
&APIVersion=1
&Date=Tue%2C%2027%20Dec%202011%2013%3A41%3A09%20GMT
&OTSAccessKeyId=ID1
&SignatureMethod=HmacSHA1
&SignatureVersion=1
&Signature=XXXXXXXXXXX
```

- 成功返回样例：

```
<?xml version="1.0" encoding=”UTF-8”?>
<AbortTransactionResult>
  <Code>OK</Code>
  <RequestID>xxxxxxxxxxxxx</RequestID>
  <HostID>xxxxxxxxxxxx</HostID>
</AbortTransactionResult>
```

查询操作

- GetRow

- 获取表或视图中一行的数据。
- 参数
  - TableName
    - 值为表名或”表名.视图名”。
    - 类型为名称。
    - 必选。
  - PK.*X*.Name
    - 值为 PK 列名，*X* 为序号，从 1 开始向上严格递增。顺序必须和建表时的 PK 列顺序一致。
    - 类型为名称。
    - 必选，至少一个，必须写入完整的表或视图的 PK 列名称，保证顺序。
    - PK.*X* 的属性写完后才能写下一个。
  - PK.*X*.Value
    - 值为对应 PK 列的值。
    - 类型为 PK.*X*.Type 所描述的类型。
    - 必选，必须和 PK.*X*.Name 一一对应。
  - PK.*X*.Type
    - 值为对应的 PK 列的类型。
    - 类型为枚举，必须为 INTEGER，STRING，BOOLEAN 之一。
    - 必选，必须和 PK.*X*.Name 一一对应。
  - Column.*X*.Name
    - 值为列名。
    - 类型为名称。
    - 可选，可重复。若指定，则查询指定的列的值，否则查询所有列的值。
  - TransactionID
    - 表示此次操作是否在一个事务中。需传入 StartTransactionAPI 生成的事务 ID。
    - 类型为合法的事务 ID。
    - 可选。
- 返回说明：不管一行是否真正存在，GetRow 逻辑上读不到数据就返回空，不抛任何异常。
  - 1. 当一行不存在，读任何字段包括 PK 字段，都会返回空的结果；
  - 2. 当一行存在，读 PK 字段会返回 PK 字段，读其他字段，如果存在则返回实际数据，否则返回空;
  - 3. 使用 GetRow 读 PK 字段或者*来判断一行是否存在，读到数据说明一行存在，否则一行不存在；
- 请求样例

```
http://service.ots.aliyun.com/GetRow?
TableName=Table1
&PK.1.Name=PK1
```

```
&PK.1.Value=%27a%27
&PK.1.Type=STRING
&PK.2.Name=PK2
&PK.2.Value=10
&PK.2.Type=INTEGER
&APIVersion=1
&Date=Tue%2C%2027%20Dec%202011%2013%3A41%3A09%20GMT
&OTSAccessKeyId=ID1
&SignatureMethod=HmacSHA1
&SignatureVersion=1
&Signature=XXXXXXXXXXX
```

- 成功返回样例

```
<?xmlversion="1.0" encoding=”UTF-8”?>
<GetRowResult>
  <Table name="Table1">
    <Row>
      <Column PK=”true”>
        <Name>PK1</Name>
        <Value type=”STRING”>a</Value>
      </Column>
      <Column PK=”true”>
        <Name>PK2</Name>
        <Value type=”INTEGER”>10</Value>
      </Column>
      <Column>
        <Name>C1</Name>
        <Value type=”STRING”>xml</Value>
      </Column>
      <Column>
        <Name>C2</Name>
        <Value type=”STRING”>html</Value>
      </Column>
    </Row>
  </Table>
  <RequestID>xxxxxxxxxxxxx</RequestID>
  <HostID>xxxxxxxxxxxx</HostID>
</GetRowResult>
```

- GetRowsByRange
  - 获取表或视图中主键的特定范围内的多行数据。
  - 参数
    - TableName
      - 值为表名或”表名.视图名”。

- 类型为名称。
- 必选。

◆ PK.*X*.Name
- 值为 PK 列名，*X* 为序号，从 1 开始向上严格递增。顺序必须和建表时的 PK 列顺序一致。
- 类型为名称。
- 必选，至少一个，必须写入表或视图的全部的 PK 列名称或从第一个 PK 列开始的连续多个 PK 列名称(至少一个 PK 列名称)，保证顺序。
- PK.*X* 的属性写完后才能写下一个。

◆ PK.*X*.Value
- 值为对应 PK 列的值。
- 类型为 PK.*X*.Type 所描述的类型。
- 除最后一个 PK 列外必选，必须和 PK.*X*.Name 一一对应。

◆ PK.*X*.Type
- 值为对应 PK 列的类型。
- 类型为枚举，必须为 INTEGER， STRING， BOOLEAN 之一。
- 除最后一个 PK 列外必选，必须和 PK.*X*.Name 一一对应。

◆ PK.*X*.RangeBegin
- 值为查询的对应 PK 列范围的开始值（范围是一个左闭右开区间），或 INF_MIN，表示无限小（如果 IsReverse 的值为 TRUE，则可以传入 INF_MAX，表示无限大）。
- 类型为 PK.*X*.RangeType 所描述的类型。
- 最后一个 PK 列必选，需和 PK 列的 Name 对应。

◆ PK.*X*.RangeEnd
- 值为查询的对应 PK 列范围的结束值（范围是一个左闭右开区间），或 INF_MAX，表示无限大（如 IsReverse 的值为 TRUE，则可以传入 INF_MIN，表示无限小）。
- 类型为 PK.*X*.RangeType 所描述的类型。
- 最后一个 PK 列必选，需和 PK 列的 Name 对应。

◆ PK.*X*.RangeType
- 值为查询的对应 PK 列范围的类型。
- 类型为枚举，必须为 INTEGER， STRING， BOOLEAN 之一。
- 最后一个 PK 列必选，需和 PK 列的 Name 对应。

◆ Column.*X*.Name
- 值为列名。
- 类型为名称。
- 可选，可重复。若指定，则查询指定的列的值，否则查询所有列的值。

◆ Top
- 表示结果返回前多少行。
- 类型为 INTEGER。
- 可选。若指定，则只返回结果集合中前面指定行数的结果。若不指定，则返回整个结果集合。

- TransactionID
  - 表示此次操作是否在一个事务中。需传入 StartTransactionAPI 生成的事务 ID。
  - 类型为合法的事务 ID。
  - 可选。
- IsReverse
  - 表示查询是否是从大到小进行读取。
  - 类型为枚举，必须为 TRUE， FALSE 之一。
  - 可选，默认值为 FALSE。
  - 当值为 FALSE 时，PK.*X*.RangeBegin 的值一定要小于 PK.*X*.RangeEnd 的值。当值为 TRUE 时，PK.*X*.RangeBegin 的值一定要大于 PK.*X*.RangeEnd 的值。

■ 请求样例

```
http://service.ots.aliyun.com/GetRowsByRange?
TableName=Table1
&PK.1.Name=PK1
&PK.1.Value=%27a1%27
&PK.1.Type=STRING
&PK.2.Name=PK2
&PK.2.RangeBegin=INF_MIN
&PK.2.RangeEnd=INF_MAX
&PK.2.RangeType=INTEGER
&Top=10
&APIVersion=1
&Date=Tue%2C%2027%20Dec%202011%2013%3A50%3A45%20GMT
&OTSAccessKeyId=ID1
&SignatureMethod=HmacSHA1
&SignatureVersion=1
&Signature=XXXXXXXXXXX
```

■ 成功返回样例

```
<?xmlversion="1.0" encoding=”UTF-8”?>
<GetRowsByRangeResult>
  <Table name="Table1">
    <Row>
      <Column PK=”true”>
        <Name>PK1</Name>
        <Value type=”STRING”>a1</Value>
      </Column>
      <Column PK=”true”>
        <Name>PK2</Name>
        <Value type=”INTEGER”>10</Value>
      </Column>
      <Column>
```

```
            <Name>C1</Name>
            <Value type="STRING">xml</Value>
          </Column>
          <Column>
            <Name>C2</Name>
            <Value type="STRING">html</Value>
          </Column>
        </Row>
        <Row>
          <Column PK="true">
            <Name>PK1</Name>
            <Value type="STRING">a1</Value>
          </Column>
          <Column PK="true">
            <Name>PK2</Name>
            <Value type="INTEGER">20</Value>
          </Column>
          <Column>
            <Name>C1</Name>
            <Value type="STRING">java</Value>
          </Column>
          <Column>
            <Name>C2</Name>
            <Value type="STRING">c#</Value>
          </Column>
        </Row>
    </Table>
    <RequestID>xxxxxxxxxxxx</RequestID>
    <HostID>xxxxxxxxxxx</HostID>
</GetRowsByRangeResult>
```

- GetRowsByOffset
  - 获取表或视图的指定偏移量开始的多行数据。要求表或视图上必须建立分页，并且指定偏移的基准位置的方式需要和建立分页的方式一致。
  - 参数
    - TableName
      - 值为表名或"表名.视图名"。
      - 类型为名称。
      - 必选。
    - Paging.*X*.Name
      - 值为分页键的名称，*X* 为序号，从 1 开始向上严格递增。顺序必须和建表时的 PK 列顺序一致。
      - 类型为名称。

- 必选，至少一个，必须写入完整的表或视图的分页键的名称，保证顺序。
- Paging.*X* 的属性写完后才能写下一个。

◆ Paging.*X*.Value
- 值为对应分页键的值。
- 类型为 Paging.*X*.Type 所描述的值。
- 必选，必须和 Paging.*X*.Name 一一对应。

◆ Paging.*X*.Type
- 值为对应分页键的类型。
- 类型为枚举，只能为 INTEGER， STRING， BOOLEAN 之一。
- 必选，必须和 Paging.*X*.Name 一一对应。

◆ Column.*X*.Name
- 值为列名。
- 类型为名称。
- 可选，可重复。若指定，则查询指定的列的值，否则查询所有列的值。

◆ Offset
- 表示以当前的分页键标识的数据区域内从哪个偏移量开始读取。
- 类型为 INTEGER。
- 必选。

◆ Top
- 表示结果返回前多少行。
- 类型为 INTEGER。
- 必选。

◆ TransactionID
- 表示此次操作是否在一个事务中。需传入 StartTransactionAPI 生成的事务 ID。
- 类型为合法的事务 ID。
- 可选。

◆ IsReverse
- 表示查询是否是从大到小进行读取。
- 类型为枚举，必须为 TRUE， FALSE 之一。
- 可选，默认值为 FALSE。

■ 请求样例

```
http://service.ots.aliyun.com/GetRowsByOffset?
TableName=Table1
&Paging.1.Name=PK1
&Paging.1.Value=%27a1%27
&Paging.1.Type=STRING
&Offset=0
&Top=10
&APIVersion=1
```

```
&Date=Tue%2C%2027%20Dec%202011%2013%3A50%3A47%20GMT
&OTSAccessKeyId=ID1
&SignatureMethod=HmacSHA1
&SignatureVersion=1
&Signature=XXXXXXXXXXX
```

- 成功返回样例

```
<?xmlversion="1.0" encoding=”UTF-8”?>
<GetRowsByOffsetResult>
  <Table name="Table1">
    <Row>
      <Column PK=”true”>
        <Name>PK1</Name>
        <Value type=”STRING”>a1</Value>
      </Column>
      <Column PK=”true”>
        <Name>PK2</Name>
        <Value type=”INTEGER”>10</Value>
      </Column>
      <Column>
        <Name>C1</Name>
        <Value type=”STRING”>xml</Value>
      </Column>
      <Column>
        <Name>C2</Name>
        <Value type=”STRING”>html</Value>
      </Column>
    </Row>
    <Row>
      <Column PK=”true”>
        <Name>PK1</Name>
        <Value type=”STRING”>a1</Value>
      </Column>
      <Column PK=”true”>
        <Name>PK2</Name>
        <Value type=”INTEGER”>20</Value>
      </Column>
      <Column>
        <Name>C1</Name>
        <Value type=”STRING”>java</Value>
      </Column>
      <Column>
        <Name>C2</Name>
        <Value type=”STRING”>c#</Value>
```

```
        </Column>
      </Row>
    </Table>
    <RequestID>xxxxxxxxxxxxx</RequestID>
    <HostID>xxxxxxxxxxxx</HostID>
</GetRowsOffsetResult>
```

更改操作

- PutData
  - 插入一行或修改指定行中的数据。
  - 参数
    - TableName
      - 值为表名，不能为视图名。
      - 类型为名称。
      - 必选。
    - PK.*X*.Name
      - 值为 PK 列名，*X* 为序号，从 1 开始向上严格递增。顺序必须和建表时的 PK 列顺序一致。
      - 类型为名称。
      - 必选，至少一个，必须写入完整的表 PK 列名称，保证顺序。
      - PK.*X* 的属性写完后才能写下一个。
    - PK.*X*.Value
      - 值为对应 PK 列的值。
      - 类型为 PK.*X*.Type 所描述的类型。
      - 必选，必须和 PK.*X*.Name 一一对应。
    - PK.*X*.Type
      - 值为对应 PK 列的类型。
      - 类型为枚举，必须为 INTEGER， STRING， BOOLEAN 之一。
      - 必选，必须和 PK.*X*.Name 一一对应。
    - Column.*X*.Name
      - 值为列名。
      - 类型为名称。
      - 可选，可重复。若指定，则插入指定的列，否则插入仅包含主键列数据的行，其他列不存在。
    - Column.*X*.Value
      - 值为对应的列的值。
      - 类型为 Column.*X*.Type 所描述的类型。
      - 若 Column.*X*.Name 指定则必选，必须和 Column.*X*.Name 一一对应。
    - Column.*X*.Type
      - 值为对应的列的类型。
      - 类型为枚举，必须为 INTEGER， STRING， BOOLEAN， DOUBLE

之一。

- 若 Column.*X*.Name 指定则必选，必须和 Column.*X*.Name 一一对应。

◆ Checking

- 表示是否做数据的存在性检查，分为不检查(No)，检查 Row 必须不存在（Insert），检查 Row 必须存在(Update)三种。
- 类型为枚举值，取值只能为: NO， INSERT， UPDATE 之中的一个（大写）。分别对应上述条目中的三种检查类型。
- 可选。若不指定，默认值为 NO。

◆ TransactionID

- 表示此次操作是否在一个事务中。需传入 StartTransactionAPI 生成的事务 ID。
- 类型为合法的事务 ID。
- 可选。

■ 请求样例

```
http://service.ots.aliyun.com/PutData?
TableName=Table1
&PK.1.Name=PK1
&PK.1.Value=%27a3%27
&PK.1.Type=STRING
&PK.2.Name=PK2
&PK.2.Value=30
&PK.2.Type=INTEGER
&Column.1.Value=%27Test%27
&Column.1.Type=STRING
&Column.2.Value=%27Demo%27
&Column.2.Type=STRING
&Checking=INSERT
&APIVersion=1
&Date=Tue%2C%2027%20Dec%202011%2011%3A25%3A44%20GMT
&OTSAccessKeyId=ID1
&SignatureMethod=HmacSHA1
&SignatureVersion=1
&Signature=XXXXXXXXXXX
```

■ 成功返回样例

```
<?xmlversion="1.0" encoding=”UTF-8”?>
<PutDataResult>
  <Code>OK</Code>
  <RequestID>xxxxxxxxxxxxx</RequestID>
  <HostID>xxxxxxxxxxxx</HostID>
</PutDataResult>
```

- DeleteData
  - 删除指定的行或行中的数据。

- 参数
  - TableName
    - 值为表名，不能为视图名。
    - 类型为名称。
    - 必选。
  - PK.*X*.Name
    - 值为 PK 列名，*X* 为序号，从 1 开始向上严格递增。顺序必须和建表时的 PK 列顺序一致。
    - 类型为名称。
    - 必选，至少一个，必须写入完整的表的 PK 列名称，保证顺序。
    - PK.*X* 的属性写完后才能写下一个。
  - PK.*X*.Value
    - 值为对应 PK 列的值。
    - 类型为 PK.*X*.Type 所描述的类型。
    - 必选，必须和 PK.*X*.Name 一一对应。
  - PK.*X*.Type
    - 值为对应 PK 列的类型。
    - 类型为枚举，只能为 INTEGER， STRING， BOOLEAN 之一。
    - 必选，必须和 PK.*X*.Name 一一对应。
  - Column.*X*.Name
    - 值为列名。
    - 类型为名称。
    - 可选，可重复。若指定，则删除指定的列的值，否则删除一整行。（不做任何检查）
  - TransactionID
    - 表示此次操作是否在一个事务中。需传入 StartTransactionAPI 生成的事务 ID。
    - 类型为合法的事务 ID。
    - 可选。
- 请求样例

```
http://service.ots.aliyun.com/DeleteData?
TableName=Table1
&PK.1.Name=PK1
&PK.1.Value=%27a3%27
&PK.1.Type=STRING
&PK.2.Name=PK2
&PK.2.Value=30
&PK.2.Type=INTEGER
&APIVersion=1
&Date=Tue%2C%2027%20Dec%202011%2011%3A25%3A44%20GMT
&OTSAccessKeyId=ID1
&SignatureMethod=HmacSHA1
&SignatureVersion=1
```

```
&Signature=XXXXXXXXXXX
```

- 成功返回样例

```
<?xmlversion="1.0" encoding=”UTF-8”?>
<DeleteDataResult>
  <Code>OK</Code>
  <RequestID>xxxxxxxxxxxxx</RequestID>
  <HostID>xxxxxxxxxxxx</HostID>
</DeleteDataResult>
```

- BatchModifyData
  - 把 PutData 和 DeleteData 的多次调用组合成一个调用。要求这个组合调用必须在事务中，且所有数据更改的操作必须操作同一个数据分片键的数据，并且这个数据分片键的值要和开始事务的数据分片键的值相同。
  - 参数
    - TableName
      - 值为表名，不能为视图名。
      - 类型为名称。
      - 必选。
    - Modify.*Y*.Type
      - 表示第 *Y* 个操作的类型，只能为 PutData 或 DeleteData。
      - 类型为枚举，值只能为：PUT， DELETE 之一（大写），分别对应上述条目的类型。
      - 可选，可重复。必须写入完整的一个 Modify 操作后才能写下一个。
    - Modify.*Y*.PK.*X*.Name
      - 表示本次 Modify 的 PK 列名，*X* 为序号，从 1 开始向上严格递增。顺序必须和建表时的 PK 列顺序一致。
      - 类型为名称。
      - 若本次 Modify 存在则必选，至少一个，必须写入完整的表或 PK 列名称，保证顺序。
      - Modify.*Y*.PK.*X* 的属性写完后才能写下一个。
    - Modify.*Y*.PK.*X*.Value
      - 表示本次 Modify 中对应 PK 列的值。
      - 类型为 Modify.*Y*.PK.*X*.Type 中所描述的类型。
      - 若本次 Modify 存在则必选，必须和 Modify.*Y*.PK.*X*.Name 一一对应。
    - Modify.*Y*.PK.*X*.Type
      - 表示本次 Modify 中对应 PK 列的类型。
      - 类型为枚举，只能为 INTEGER， STRING， BOOLEAN 之一。
      - 若本次 Modify 存在则必选，必须和 Modify.*Y*.PK.*X*.Name 一一对应。
    - Modify.*Y*.Column. *X*.Name
      - 表示本次 Modify 的列名。

- 类型为名称。
- 可选，可重复。若指定，当 Type 为 Put 时，插入指定列的值；当 Type 为 Delete 时，删除指定的值。若不指定，当 Type 为 Put 时，插入一个空行；当 Type 为 Delete 时，删除一整行。

◆ Modify.*Y*.Column.*X*.Value
- 表示本次 Modify 的列的值。
- 类型为 Modify.*Y*.Column.*X*.Type 所描述的类型。
- 可选，且只有本 Modify 的 Type 为 Put 时才能指定这个参数。

◆ Modify.*Y*.Column.*X*.Type
- 表示本次 Modify 的列的类型。
- 类型为枚举，只能为 INTEGER， STRING， BOOLEAN， DOUBLE 之一。
- 可选，且只有本 Modify 的 Type 为 Put 时才能指定这个参数。

◆ Modify.*Y*.Checking
- 表示 Put 操作的存在性检查方式。具体见 API PutData 的 Checking 参数。
- 类型为枚举值，取值只能为: NO， INSERT， UPDATE 之中的一个（大写）。
- 可选，且只有本 Modify 的 Type 为 Put 时才能指定这个参数。若不指定，默认值为 NO。

◆ TransactionID
- 需传入 StartTransactionAPI 生成的事务 ID。
- 类型为合法的事务 ID。
- 必选。

■ 请求样例

```
http://service.ots.aliyun.com/BatchModifyData?
TableName=Table1
&Modify.1.Type=DELETE
&Modify.1.PK.1.Name=PK1
&Modify.1.PK.1.Value=%27a%27
&Modify.1.PK.1.Type=STRING
&Modify.1.PK.2.Name=PK2
&Modify.1.PK.2.Value=10
&Modify.1.PK.2.Type=INTEGER
&Modify.2.Type=DELETE
&Modify.2.PK.1.Name=PK1
&Modify.2.PK.1.Value=%27a%27
&Modify.2.PK.1.Type=STRING
&Modify.2.PK.2.Name=PK2
&Modify.2.PK.2.Value=20
&Modify.2.PK.2.Type=INTEGER
&Modify.3.Type=PUT
&Modify.3.PK.1.Name=PK1
```

```
&Modify.3.PK.1.Value=%27a%27
&Modify.3.PK.1.Type=STRING
&Modify.3.PK.2.Name=PK2
&Modify.3.PK.2.Value=30
&Modify.3.PK.2.Type=INTEGER
&Modify.3.Column.1.Name=C1
&Modify.3.Column.1.Value=%Test%27
&Modify.3.Column.1.Type=STRING
&Modify.3.Column.2.Name=C2
&Modify.3.Column.2.Value=%27Demo%27
&Modify.3.Column.2.Type=STRING
&TransactionID=GgAAAG90c19mdF90ZXN0X2NvbnRhY3RfdGVzdDEzGwAAADEzNDQ
5ODEzMTItMTMyNDk4NTI1NzMzOTE3OAMFAAAAdWlkLTH9Vpqa
&APIVersion=1
&Date=Tue%2C%2027%20Dec%202011%2011%3A30%3A46%20GMT
&OTSAccessKeyId=ID1
&SignatureMethod=HmacSHA1
&SignatureVersion=1
&Signature=XXXXXXXXXXX
```

- 成功返回样例

```
<?xmlversion="1.0" encoding=”UTF-8”?>
<BatchModifyDataResult>
   <Code>OK</Code>
   <RequestID>xxxxxxxxxxxxx</RequestID>
   <HostID>xxxxxxxxxxxx</HostID>
</BatchModifyDataResult>
```

错误信息

任何一个 Request 出错后，返回一个包含错误信息的 XML。样例:

```
<?xmlversion="1.0" encoding=”UTF-8”?>
<Error>
   <Code>PartitionNotFound</Code>
   <Message>Partition xxxx-xxxx-xxxx-xxxx is not found.</Message>
   <RequestID>xxxxxxxxxxxxx</RequestID>
   <HostID>xxxxxxxxxxxx</HostID>
</Error>
```

各错误的错误代码和错误信息以及表述的含义见下。

一般错误

身份认证错误

ErrorCode	ErrorMsg	描述
OTSAuthFailed	Missing parameter: {Parameter}	请求中缺失某项与用户认证相关的参数，这些参数包括 Date，OTSAccessKeyId，Signature，SignatureMethod，SignatureVersion，缺其中任何一项均不可。
OTSAuthFailed	Date header bad format: {Date}	Date字段未采用协议规定的格式
OTSAuthFailed	The date has expired: {Date}	请求的Date已经过期
OTSAuthFailed	Unsupport signature method: {SignatureMethod}	请求指定了不支持的签名方法
OTSAuthFailed	Unsupport signature version: {SignatureVersion}	请求指定了不支持的签名版本
OTSAuthFailed	User signature dose not match.	用户请求中包含的签名与服务器计算的签名不一致
OTSAuthFailed	This accessId has been disabled.	请求中使用的AccessId已经被禁用
OTSAuthFailed	AccessId not exist.	请求中包含的AccessId在系统中不存在

一般请求错误

ErrorCode	ErrorMsg	描述
OTSUnsupportOperation	Unsupport operation : {URI}	请求中包含错误的API URI
OTSParameterInvalid	Only POST and GET method for request is supported.	请求使用了错误的HTTP request method
OTSMissingParameter	The request must contain the parameter of APIVersion	请求中未包含APIVersion参数

OTSParameterInvalid	Failed to get arguments from post data, maybe the size of post data is too large or something unexpectable happened while reading	请求通过POST传输，请求中POST的数据超过最大上限或者读取POST数据时意外中断
OTSParameterInvalid	Invalid API version, we only support version 1 currentlly.	请求中包含不支持的API Version
OTSParameterInvalid	Request contains none UTF-8 characters.	用户请求中包含非UTF-8字符
OTSParameterInvalid	Unrecognizable parameter {ParameterName}	请求中包含协议外的参数或者参数的相对顺序错误导致服务器无法识别
OTSParameterInvalid	Unsupported data type: {TypeName}	不支持的列类型
OTSParameterInvalid	Can't convert {Value} to type {Type}	无法将用户请求中包含的参数值转换为用户指定的类型
OTSInternalServerError	{Server error message}	内部错误，如有疑问请联系客服

计量相关错误

ErrorCode	ErrorMsg	描述
OTSQuotaExhausted	Your quota has exhausted.	用户的Quota限额已经用满

API 特定错误

CreateTableGroup

ErrorCode	ErrorMsg	描述
OTSMissingParameter	The request must contain the parameter of {parameter name}	请求中缺失某项必需的参数
OTSParameterInvalid	{PartitionKeyType} is an invalid type for primary key.	PartitionKeyType的值不正确
OTSParameterInvalid	Value({TableGroupName}) for table/tableGroup name is invalid.	表组/表名称不符合规范

DeleteTableGroup

ErrorCode	ErrorMsg	描述

OTSMissingParameter	The request must contain the parameter of {parameter name}	请求中缺失某项必需的参数
OTSParameterInvalid	Value({TableGroupName}) for table/tableGroup name is invalid.	表组/表名称不符合规范

CreateTable

ErrorCode	ErrorMsg	描述
OTSMissingParameter	The request must contain the parameter of {parameter name}	请求中缺失某项必需的参数
OTSParameterInvalid	Value({TableName}) for table/tableGroup name is invalid.	表或者表组名称不符合规范
OTSParameterInvalid	Value({ViewName}) for table/tableGroup name is invalid.	视图名不符合规范
OTSParameterInvalid	Value({ColumnName}) for column name is invalid	表或视图中的列名不符合规范
OTSParameterInvalid	{PKType} is an invalid type for primary key.	表的主键类型不正确
OTSParameterInvalid	{ColumnType} is an invalid type for column.	表或视图中的列类型不正确
OTSParameterInvalid	The name of primary key must be unique with each other.	表中PK列名不唯一
OTSParameterInvalid	There must be at least one primary key in table.	表中未指定任何PK列
OTSParameterInvalid	The value of PagingKeyLen must be less than length of table's primary key list.	PagingKeyLen参数不正确，必须小于表中PK列长度
OTSParameterInvalid	The name of view must be unique with each other.	表的多个视图名称不唯一
OTSParameterInvalid	The first primary key of view must be the same with table's partition key.	视图的数据分片键与表的数据分片键不一致
OTSParameterInvalid	The name of primary key and column in view must be unique with each other.	视图中的PK列或者属性列的名称不唯一
OTSParameterInvalid	There must be at least one primary key in view.	视图中未指定任何PK列
OTSParameterInvalid	The value of PagingKeyLen must be less than length of view's primary key list.	PagingKeyLen参数不正确，必须小于视图中PK列长度

GetTableMeta

ErrorCode	ErrorMsg	描述
OTSMissingParameter	The request must contain the parameter of {parameter name}	请求中缺失某项必需的参数
OTSParameterInvalid	Value({TableName}) for table/tableGroup name is invalid.	表不符合规范

DeleteTable

ErrorCode	ErrorMsg	描述
OTSMissingParameter	The request must contain the parameter of {parameter name}	请求中缺失某项必需的参数
OTSParameterInvalid	Value({TableName}) for table/tableGroup name is invalid.	表名不符合规范

GetRow

ErrorCode	ErrorMsg	描述
OTSMissingParameter	The request must contain the parameter of {parameter name}	请求中缺失某项必需的参数
OTSParameterInvalid	Value({TableName}) for table/tableGroup name is invalid.	若读取表，表名不符合规范
OTSParameterInvalid	Value({TableName}) for view name is invalid.	若读取视图，视图名不符合规范
OTSParameterInvalid	Value({ColumnName}) for column name is invalid	列名不符合规范
OTSParameterInvalid	{PKType} is an invalid type for primary key.	PK列的类型不正确
OTSParameterInvalid	PartitionKey is not within this Transaction.	该行不存在于事务的数据分片键所指定的数据范围内
OTSMetaNotMatch	Primary key meta from request not match with table meta.	PK列相关信息与表结构中有关信息不匹配
OTSParameterInvalid	Request must indicate primary key.	请求中未指定任何PK列
OTSParameterInvalid	TransactionID is invalid.	事务ID不合法

GetRowsByRange

ErrorCode	ErrorMsg	描述

OTSMissingParameter	The request must contain the parameter of {parameter name}	请求中缺失某项必需的参数
OTSParameterInvalid	Value({TableName}) for table/tableGroup name is invalid.	表名不符合规范
OTSParameterInvalid	Value({TableName}) for view name is invalid.	视图名不符合规范
OTSParameterInvalid	Value({ColumnName}) for column name is invalid	列名不符合规范
OTSParameterInvalid	{PKType} is an invalid type for primary key.	PK列的类型不正确
OTSParameterInvalid	PartitionKey is not within this Transaction.	请求的数据不存在于事务的数据分片键所指定的数据范围内
OTSMetaNotMatch	Primary key meta from request not match with table meta.	PK列的有关信息与表中的相关信息不匹配
OTSParameterInvalid	TransactionID is invalid.	事务ID不合法
OTSParameterInvalid	Request must set at least one primary key if in transaction.	指定了事务ID，却没有指定数据分片键

PutData

ErrorCode	ErrorMsg	描述
OTSMissingParameter	The request must contain the parameter of {parameter name}	请求中缺失某项必需的参数
OTSParameterInvalid	Value({TableName}) for table/tableGroup name is invalid.	表名不符合规范
OTSParameterInvalid	Value({ColumnName}) for column name is invalid	列名不符合规范
OTSParameterInvalid	{ColumnType} is an invalid type for column.	列类型不正确
OTSParameterInvalid	{PKType} is an invalid type for primary key.	PK列的类型不正确
OTSParameterInvalid	Request must indicate primary key.	请求中未指定PK列值
OTSParameterInvalid	PartitionKey is not within this Transaction.	请求的数据不存在于事务的数据分片键所指定的数据范围内
OTSMetaNotMatch	Primary key meta from request not match with table meta.	PK列的相关信息与表结构中的相

		关信息不匹配
OTSParameterInvalid	TransactionID is invalid.	事务ID不合法

DeleteData

ErrorCode	ErrorMsg	描述
OTSMissingParameter	The request must contain the parameter of {parameter name}	请求中缺失某项必需的参数
OTSParameterInvalid	Value({TableName}) for table/tableGroup name is invalid.	表名不符合规范
OTSParameterInvalid	Value({ColumnName}) for column name is invalid	列名不符合规范
OTSParameterInvalid	{ColumnType} is an invalid type for column.	列类型不正确
OTSParameterInvalid	{PKType} is an invalid type for primary key.	PK列的类型不正确
OTSParameterInvalid	Request must indicate primary key.	请求中未指定PK列值
OTSParameterInvalid	PartitionKey is not within this Transaction.	请求的数据不存在于事务的键所指定的数据范围内
OTSMetaNotMatch	Primary key meta from request not match with table meta.	PK列的相关信息与表结构中的相关信息不匹配
OTSParameterInvalid	TransactionID is invalid.	事务ID不合法

BatchModifyData

ErrorCode	ErrorMsg	描述
OTSMissingParameter	The request must contain the parameter of {parameter name}	请求中缺失某项必需的参数
OTSParameterInvalid	Value({TableName}) for table/tableGroup name is invalid.	表名不符合规范
OTSParameterInvalid	Value({ColumnName}) for column name is invalid	列名不符合规范
OTSParameterInvalid	{ColumnType} is an invalid type for column.	列类型不正确
OTSParameterInvalid	{PKType} is an invalid type for primary key.	PK列的类型不正确
OTSParameterInvalid	Request must indicate primary key.	请求中未指定PK列值
OTSParameterInvalid	PartitionKey is not within this Transaction.	请求的数据不存在于事务的键所

		指定的数据范围内
OTSMetaNotMatch	Primary key meta from request not match with table meta.	PK列的相关信息与表结构的相关信息不匹配
OTSParameterInvalid	TransactionID is invalid.	事务ID不合法
OTSParameterInvalid	The first primary key of each item in BatchModifyData must be the same with transaction's partition key	所有的BatchModify Item未包含相同的数据分片键

OTS 存储相关异常

ErrorCode	ErrorMsg	描述
OTSStoragePartitionNotReady	The partition has not been loaded.	数据分片还未准备好
OTSStorageTimeout	Operation timeout.	超时错误，操作结果未定义
OTSStorageObjectNotExist	Requested table/view doesn't exist.	对象不存在，比如表不存在，视图不存在
OTSStorageObjectAlreadyExist	Requested table/view does exist.	要创建的对象已经存在，比如数据库，表已经存在
OTSStoragePrimaryKeyAlreadyExist	Row to insert does exist.	要插入的行已经存在
OTSStoragePrimaryKeyNotExist	Row to update doesn't exist.	要更新的行不存在
OTSStorageSessionNotExist	Session timeout.	要操作的事务已经过期。
OTSStorageTxnLockKeyFail	Transaction timeout because cannot accquire exclusive lock.	不能获取指定的事务锁，一般是因为锁竞争
OTSStorageServerBusy	Service is busy.	服务器忙，请等待
OTSStorageViewIncompletePK	Cannot build index because required index pk column lost.	视图的PK列不完整，OTS要求主表的PK列和视图的PK列都要完整，不允许null存在
OTSStorageInvalidPK	Column-number or type of primary key does not match the one defined in table	无效PK列，比如类型不对

	schema.	
OTSStorageDataOutOfRange	Can not access: data are out of range. Usually this error occurs when: 1. Access data which are out of data store's range. 2. Access data which are out of txn's range.	数据超出数据分片范围
OTSStoragePrimaryKeyInvalid	The primary key's format is not consistent with the one defined in the meta.	无效PK列，比如类型不对
OTSStorageParameterInvalid	The input parameter is invalid.	参数无效，比如类型不对
OTSStorageInvalidColumnName	Column name include invalid character or column name equal with primary key column.	列名中包含无效字符，或与PK列名同名。
OTSStorageUnknownError	Unknown error.	未知错误，操作结果未定义，如有疑问请联系客服
OTSStorageInternalError	{Server error message}	内部错误，操作结果未定义，如有疑问请联系客服