**Project 1-tbl API Document**

|  |  |
| --- | --- |
| **Date** | **06.05.2013** |
| **Last modified by** |  |

# Introduction

What the web service does:

It allows us to:

1. Get a list of all elements in one table (in project) and to filter that list (filtering coming later).
2. Get info on a specific element.
3. Get the number of elements in one table(in project).
4. Add element to table(in project).
5. Modify element in table (in project).
6. Delete element from table (in project).

## Change Management

|  |  |  |
| --- | --- | --- |
| Date | Author | Changes |
| 02.05.2013 | Fareed | Started writing the document |
| 06.05.2013 | Roxan&Aziz&Fareed | Building requests and responses |
| 13.05.2013 | Aziz & Fareed |  |
| 23.05.2013 | Aziz | Changed the HTTP method of Delete to POST |

# API's

## General Requests

### Method

API requests are performed using either the GET/POST/PUT/DELETE methods, in the following format:

http://<server\_name>/<Base URI>/<api\_method>?<JSON format >

Whereas:

* *<server\_name> is the 1-tbl server host address*
* *<api\_method> is the API we are invoking*
* *?<JSON format > are the JSON formatted parameters for the API call*

### Global Parameters

All JSON requests must contain the following parameters in the HTTP method:

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| **Version** | String | Version of the 1-TBL |
| **Token** | String | Unique identifier of the user |
| **Project** | String | Name of the project |
| **Table** | String | Name of the Table |

## General Responses

All API responses are in the following JSON format:

|  |
| --- |
| {  "status":"<Boolean\_status>",  "returnedValue": "<returned>"  } |

**Whereas**:

* *< Boolean\_status > - true or false*
* *<returned> - returned value or error code*

Example: { "status":false,"returnedValue": "2"}

### Global Parameters

All API responses must contain the following parameters in the JSON reply:

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| **status** | Boolean | Boolean indicating whether the call was successful or not |
| **returnedValue** | String | Returned value as string , in case of error , it will indicate the corresponding error code(See table [below)](#h.2s8eyo1) |

## Select API

**API Method:** Select

HTTP Method : GET

**JSON Parameters:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| **Name** | **Type** | **Description** |
| **columns** | Array | List of columns to be selected, separated by the ; between them |
| **name** | String | Name of the column in the table |
| **whereClause** | String | Conditions to be put in the where clause in the query |
| **aggregate** | Aggregation | Aggregation object containing the column name and the aggregation function to be used |

### 

Example: http://1tbl.com/api/select?query={"version”:”v1.0”,”token":"bewcwec78we","project":"test","table":"test\_table","columns":["ID","Age","Height"],"whereClause":"(ID < 5 and Height > 180 ) or age > 13"}

JSON sent:

{

"version”:”v1.0”,

”token":"bewcwec78we",

"project":"test","table":"test\_table",

"columns":["ID","Age","Height"],

"whereClause":"(ID < 5 and Height > 180 ) or age > 13"

“aggregate”:

“columnName” :”Age”

“AGGTYPE” : “COUNT”

}

\* if the aggregate is empty - then no aggregate function is applied

In the above example:

* Select rows from the table , displaying only ID , Age and Height , according to the conditions given in the whereClause

### Notes

* If the conditions section was missing then it is considered as a select all rows query displaying specified columns.
* If the columns section was missing then it is considered as select all columns query.

### Response Options

|  |
| --- |
| {  "status":true,  "returnedValue": [  {"column1":"value1","column2":"value2"},  {"column1":"value3","column2":"value4"}  ]  } |

***Whereas****:*

*Returned value is a list of rows, each entry represents and individual row, with the corresponding columns.*

|  |
| --- |
| {  "status":false,  "returnedValue": "<[ErrorCode>](#h.2s8eyo1)"  } |

## Select API

**API Method:** Select with Aggregate Function(Overloaded)

HTTP Method : GET

**JSON Parameters:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| **columns** | Array | List of columns to be selected, separated by the ; between them |
| **name** | String | Name of the column in the table |
| **whereClause** | String | Conditions to be put in the where clause in the query |
| **aggObj** | Aggregation | Aggregation object containing the column name and the aggregation function to be used |

### 

Example: http://1tbl.com/api/select?query={"version”:”v1.0”,”token":"bewcwec78we","project":"test","table":"test\_table","columns":["ID","Age","Height"],"whereClause":"(ID < 5 and Height > 180 ) or age > 13"}

JSON sent:

{

"version”:”v1.0”,

”token":"bewcwec78we",

"project":"test","table":"test\_table",

"columns":["ID","Age","Height"],

"whereClause":"(ID < 5 and Height > 180 ) or age > 13"

“aggObj”:”

}

In the above example:

* Select rows from the table , displaying only ID , Age and Height , according to the conditions given in the whereClause

### Notes

* If the conditions section was missing then it is considered as a select all rows query displaying specified columns.
* If the columns section was missing then it is considered as select all columns query.

### Response Options

|  |
| --- |
| {  "status":true,  "returnedValue": [  {"column1":"value1","column2":"value2"},  {"column1":"value3","column2":"value4"}  ]  } |

***Whereas****:*

*Returned value is a list of rows, each entry represents and individual row, with the corresponding columns.*

|  |
| --- |
| {  "status":false,  "returnedValue": "<[ErrorCode>](#h.2s8eyo1)"  } |

## 

## Insert API

**API Method:** Insert

HTTP Method : POST

**Parameters:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| **columns** | Array | The entry columns |
| **name** | String | Name of the column in the given index |
| **values** | Array | The values to be assigned respectively with the columns order. |
| **value** | String | Value of the column in the given index |

Example: [http://1tbl.com/api/insert](http://1tbl.com/insert)

JSON sent:

{

“version”:”v1.0”,

"token":"bewcwec78we",

"project":"test",

"table":"test\_table",

"columns":["ID","Age","Name","Nickname","Manager"],

"values":[2301390,13,"Hasan","The Boss","Mohammad Gh"]

}

In the above example:

* Inserts a row to a table.
* Assign 2301390 to 'ID' , 13 to Age , Hasan to 'Name', The Boss to 'Nickname' , Mohammad Gh to 'Manager' .

### Response Options

|  |
| --- |
| {  "status":true  } |

|  |
| --- |
| {  "status":false,  "returnedValue": "<[ErrorCode>](#h.2s8eyo1)"  } |

## Update API

HTTP Method : PUT

**API Method:** Update

**Parameters:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| **whereClause** | String | Conditions for the row to be selected by , and the updated. |
| **setColumns** | Array | List of columns to be used in the 'set' operation, |
| **name** | String | Name of the column |
| **setValues** | Array | The values to be set for the columns given |
| **value** | String | Value to be set to the column with the corresponding index in the setColumns array. |

### Separation Operators

The user may separate the operations into groups, by adding parenthesis around each group, and the proper operator between them

Example:

[http://1tbl.com/api/update](http://1tbl.com/update?%7b%22token%22:%22bewcwec78we%22,%22project%22:%22test%22,%22table%22:%22test_table)

JSON sent:

{

“version”:”v1.0”,

"token":"bewcwec78we",

"project":"test",

"table":"test\_table",

"whereClause":"ID > 20 or Age < 40",

"setColumns":["ID","Age","Name","Nickname","Manager"],

"setValues":[2301390,13,"Hasan","The Boss","Mohammad Gh"]

}

In the above example:

* Updates a row in the table according to the given whereClause
* Set 2301390 to 'ID' , 13 to Age , Hasan to 'Name', The Boss to 'Nickname' , Mohammad Gh to 'Manager' .

### Note

If the conditions section was missing then it is considered as a update all query.

### Response Options

|  |
| --- |
| {  "status":true,  "returnedValue":"<number>"  } |

***Whereas****:*

*<number>: indicates the number of rows affected in table.*

*If no such row then returnedValue is 0.*

|  |
| --- |
| {  "status":true,  "returnedValue": "<[ErrorCode>](#h.2s8eyo1)"  } |

## Delete API

HTTP method : POST

**API Method:** Delete

**Parameters:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| **whereClause** | String | Conditions to be put in the where clause in the query |

### Separation Operators

The user may separate the operations into groups, by adding parenthesis around each group, and the proper operator between them

Example: http://1tbl.com/api/delete

JSON sent:

{

“version”:”v1.0”,

"token":"bewcwec78we",

"project":"test",

"table":"test\_table",

whereClause":"(ID < 5 and Height > 180 ) or age > 13"

}

In the above example:

* Deletes rows from the table
* Where value1 equals to option1 and value2 equals to option2 or value3 equals to option3.

### Note

If the params section was missing then it is considered as a delete all rows query.

### Response Options

|  |
| --- |
| {  "status":true,  "returnedValue":"<number>"  } |

***Whereas****:*

*<number>: indicates the number of rows affected in table.*

*If no such row then returnedValue is 0.*

|  |
| --- |
| {  "status":false,  "returnedValue": "<[ErrorCode>](#h.2s8eyo1)"  } |

## Error Codes Table

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| **1** | InvalidCredentials |
| **2** | Column name not found |
| **3** | Number Format Exception – Unable to parse parameter value to the correct type |
| **4** | Illegal Condition |
| **5** | No such version |