CRM+ Querying

Introduction

The IVR Object CRM+ Query is used to query the CRM+ system using the REST API.

The queries which are used are very generic and involve writing actual SQL statements.

The advantage is that you can retrieve just about anything.

However, it does involve a little exploratory work finding out how the database is structured first.

Postman

In order to find out what tables and fields are available, we have provided a postman collection, which can be downloaded here. This simplifies testing greatly.

Collection Download

Variables

First of all, edit the collection, and setup the required variables. These are the settings you retrieved in this step here: CRM+ First Steps.

You will need to modify the variables:

- **accessKey** (from the CRM+ account)
- **userName** (the login name of the CRM+ account)
- **baseURL** (the URL of your CRM+ instance)

CRM+ Watch 0

Authorization Pre-request Script Tests Variables

These variables are specific to this collection and its requests. Learn more about collection variables.

	VARIABLE	INITIAL VALUE	CURRENT VALUE
☑	accessKey	xxxxxxxxxxxxxxx	xxxxxxxxxxxxxxx
☑	token		
☑	md5Hash		
☑	userName	mycrmaccount	mycrmaccount
☑	sessionName		
☑	userID		
☑	userId		
☑	baseURL	https://mycrm.brain-app.com	https://mycrm.brain-app.com
	Add a new variable		

Functions

The postman collection provides the following functions:

- CRM+
  - GET 1. Getchallenge
  - POST 2. Login
  - GET 3. Listtypes
  - GET 4. Query

Scripts

The postman collection contains scripts, which parse the result and update variables in the collection so the next function can execute.

The scripts can be found in the tests section:

GET {{baseURL}}/webservice.php?operation=getchallenge&username={{userName}}

Params Authorization Headers (6) Body Pre-request Script Tests Settings

```
var responseJson = pm.response.json();
console.log( "response is ", responseJson );

var token = responseJson.result.token;
pm.collectionVariables.set( "token", token );

console.log( "token is ", token );
var accessKey = pm.collectionVariables.get("accessKey").toString();

console.log( "accessKey is ", accessKey );
var md5Hash = CryptoJS.MD5(token + accessKey).toString();

console.log( "md5Hash is ", md5Hash );
pm.collectionVariables.set( "md5Hash", md5Hash );
```

Exploring the Structure of CRM+

Retrieving all Table Names

To retrieve all table names, you will need to execute the REST functions:

- 1. Getchallenge
- 2. Login
- 3. Listtypes

This will give you a JSON result containing all of the table names you can access.

For example:

```
{
    "success": true,
    "result": {
    "types": [
        "Leads",
        "Accounts",
        "Contacts",
        ...
```

Finding out the Field Names

The simplest way to find out all the field names in a particular table is to execute a **SELECT * FROM <TableName>;** query.

The sample query does this on the Contacts table.

To run this in Postman execute the following functions:

- 1. Getchallenge
- 2. Login
- 4. Query

Practical Queries

Obviously a practical query will depend entirely on what you are trying to achieve.

For example, to retrieve all contacts with a particular telephone number, you might do this:

```
# Find a contact by telephone number
SELECT * FROM Contacts WHERE phone = '%2B$caller';
```

Note, the plus sign is encoded as %2B. Here we use the variable **$caller**, which contains the caller ID without a plus in fully qualified E.164 format, to pass in the caller number to the CRM system.

Building URLs

IDs in CRM+

CRM+ returns IDs in the REST API using a combination of the Module in which the record is hosted, and the ID itself.

For example:

```
            "email": "[illegible]",
            "fax": "[illegible]",
            "firstname": "[illegible]",
            "lastname": "[illegible]",
            "mailingcity": "[illegible]",
            "mailingcountry": "Deutschland",
            "mailingstreet": "[illegible]",
            "mailingzip": "[illegible]",
            "mobile": "",
            "modifiedby": "19x4570",
            "modifiedtime": "2015-07-16 16:01:21",
            "otherphone": "",
            "partner_id": "",
            "phone": "[illegible]",
            "record_currency_id": "52x23265",
            "contact_id": "",
            "salutationtype": "Mr.",
            "assigned_user_id": "19x9",
            "title": "[illegible]",
            "yahooid": "",
            "id": "4x9"
        }
    ]
}
```


This record is in module 4, and has ID 9.

The module can be retrieved with the function **3. Listtypes**.

Here is the snippet returned by the contacts module:


```
"Contacts": {
    "isEntity": true,
    "label": "Contacts",
    "singular": "Contact",
    "createable": true,
    "updateable": true,
    "deleteable": true,
    "retrieveable": true,
    "idPrefix": "4"
},
```

Extracting the Record ID Only

In the jtel IVR script editor, this involves extracting the right hand side of the id, which can be achieved using the IVR object String functions

Building the URL

Building an URL for use in the jtel system involves the following components:

Component	Contents
baseURL	The base URL of the CRM+ instance.
module	The name of the module in which the record is present.
ID	The ID of the record without the module prefix.

The URL has the following format:

```
{{baseURL}}/index.php?action=DetailView&module={{module}}&record={{ID}}
```

For example, to display the contact with ID 1234, the following URL could be used:

```
https://mycompany.brain-app.com/index.php?action=DetailView&module=Contacts&record=1234
```