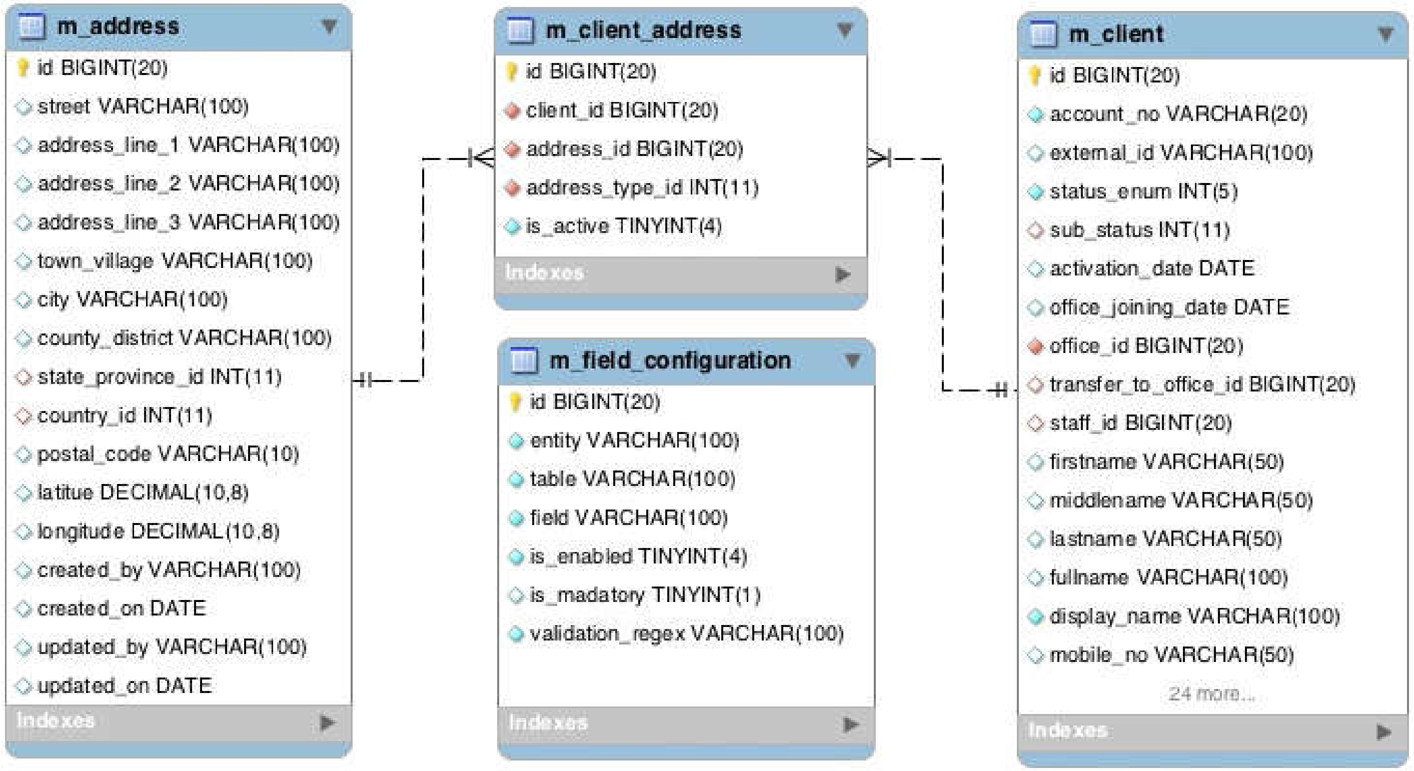
Projects / Mifos Credit bureau Integration. (Risk calibration Module -RCM) / Technical Description of Address API

Last modified Jul 06, 2016

Requirement

This technical solution document is prepared based on the user requirement documentation at Address

Schema diagram:



m_address
id BIGINT(20)
street VARCHAR(100)
address_line_1 VARCHAR(100)
address_line_2 VARCHAR(100)
address_line_3 VARCHAR(100)
town_village VARCHAR(100)
city VARCHAR(100)
county_district VARCHAR(100)
state_province_id INT(11)
country_id INT(11)
postal_code VARCHAR(10)
latitue DECIMAL(10,8)
longitude DECIMAL(10,8)
created_by VARCHAR(100)
created_on DATE
updated_by VARCHAR(100)
updated_on DATE
Indexes

m_client_address
id BIGINT(20)
client_id BIGINT(20)
address_id BIGINT(20)
address_type_id INT(11)
is_active TINYINT(4)
Indexes

m_field_configuration
id BIGINT(20)
entity VARCHAR(100)
table VARCHAR(100)
field VARCHAR(100)
is_enabled TINYINT(4)
is_madatory TINYINT(1)
validation_regex VARCHAR(100)
Indexes

m_client
id BIGINT(20)
account_no VARCHAR(20)
external_id VARCHAR(100)
status_enum INT(5)
sub_status INT(11)
activation_date DATE
office_joining_date DATE
office_id BIGINT(20)
transfer_to_office_id BIGINT(20)
staff_id BIGINT(20)
firstname VARCHAR(50)
middlename VARCHAR(50)
lastname VARCHAR(50)
fullname VARCHAR(100)
display_name VARCHAR(100)
mobile_no VARCHAR(50)
24 more...
Indexes

/ /

e for
r this

parameter.

Further the address structure is also configurable. The fields present in the address table can be enabled/disabled based on the institutions requirement. This can be done via m_field configuration table. You can mark the required fields of address table as enabled from this table.

As a part of address module, there would be a set of new APIs and also need to modify set of client related APIs to accommodate address details.

New APIs

1. POST: clients/<client_id>/address
2. GET: /clients/<client id>/address (clients all address)
3. GET /clients/<client id>/address/<address typ id>
4. GET clients/<client id>/address/<add typ id>?command=active>
5. **PUT** clients/<client id>/address/<add typ Id>

Modify client related APIs

1. POST: clients/ Proposed modifications: takes additional address data
2. GET : clients/<id> Proposed modifications: Along with client data, fetches client's addresses
3. PUT : clients/<client id> Proposed modifications: takes additional address data

⚠ if address is enabled in the global configuration, the Client API would then enforce the address to be mentioned right at the onset of creation of client. In case, the address field is not enabled this enforcement would not come into action.

New REST API for adding address of existing clients:

Please note that I have added an additional field 'is_active' to support address change of same type of address(and also to keep track of past address for audit purpose).

However this would add more complexity to the design, so in initial phase we would be accommodating it, only if time permits.

/ /

Sample Request:

POST /clients/<client id>/address

```
Content-Type: application/json
Request Body:
{
    "address_type": 1,
    "street": "507 west street",
    "address_line_1": "apt no 5",
    "address_line_2": "opposite to college park",
    "city":"Dallas",
    "state_province_id":1,
    "country_id":2,
    "postal_code":"40041",
}
```

2) **Retrieve all addresses of all types for particular client:**

GET https://DomainName/api/v1/clients/<client id>/address/

Sample Response:

GET /clients/<client id>/address

```
Content-Type: application/json
Response Body:
[
    {
        "street": "307 west street",
        "address_line_1": "apt no 5",
        "address_line_2": "opposite to college park",
        "city":"Dallas",
        "state_province_id":1,
        "country_id":2,
        "postal_code":"40041",
        "is_active":1
    },
    {

        "clientId":1,
        "addressId":3,
        "address_type": 1,
        "street": "100 west street",
        "address_line_1": "apt no 7",
        "address_line_2": "opposite to college park",
        "city":"Dallas",
        "state_province_id":1,
        "country_id":2,
        "postal_code":"40041",
        "is_active":0
    }

]
```

3). **Retreive all addresses of particular type for specified client**

/ /

Sample Response:

GET /clients/<client id>/address/<address typ id>

```
Content-Type: application/json
Response Body:
[
    {
        "clientId":1,
        "addressId":2,
        "address_type": 1,
        "street": "507 west street",
        "address_line_1": "apt no 5",
        "address_line_2": "opposite to college park",
        "city":"Dallas",
        "state_province_id":1,
        "country_id":2,
        "postal_code":"40041",
        "is_active":1
    },
    {

        "clientId":1,
        "addressId":2,
        "address_type": 1,
        "street": "100 west street",
        "address_line_1": "apt no 7",
        "address_line_2": "opposite to college park",
        "city":"Dallas",
        "state_province_id":1,
        "country_id":2,
        "postal_code":"40041",
      /                                              /
]
```

4) **Retrieve active address of particular type for specified clientid**

GET https://DomainName/api/v1/clients/<client id>/address/<add typ Id>?command=active

Sample Response:

GET clients/<client id>/address/<add typ id>?command=active

```
Content-Type: application/json
Response Body:
{
        "clientId":1,
        "addressId":2,
        "address_type": 1,
        "street": "507 west street",
        "address_line_1": "apt no 5",
        "address_line_2": "opposite to college park",
        "city":"Dallas",
        "state_province_id":1,
        "country_id":2,
        "postal_code":"40041",
        "is_active":1
}
```

5) **Update Address**

/ /

PUT https://DomainName/api/v1/clients/<client id>/address/<add typ Id>

Sample Request:

PUT clients/<client id>/address/<add typ Id>

```
Content-Type: application/json
Request Body:
{
"is_active":false
}
```

supported fields for update request:

address line fields, *addresstypeid* and *is_active*

Modify current client APIs for adding address when creating/updating clients:

When address is enabled in global configuration, creating clients would also require address attribute to be added

1) **Create Client**:

POST https://DomainName/api/v1/clients

Sample Request:

POST clients

```
Content-Type: application/json
Request Body:
```

```
"lastname": "Yton",
"externalId": "786YYH7",
"dateFormat": "dd MMMM yyyy",
"locale": "en",
"active": true,
"activationDate": "04 March 2009",
"submittedOnDate":"04 March 2009",
"savingsProductId" : 4
"address":
            [
                {
                        "clientId":1,
                        "addressId":2,
                        "address_type": 1,
                        "street": "507 west street",
                        "address_line_1": "apt no 5",
                        "address_line_2": "opposite to college park",
                        "city":"Dallas",
                        "state_province_id":1,
                        "country_id":2,
                        "postal_code":"40041",
                        "is_active":1
               },
               {

                        "clientId":1,
                        "addressId":3,
                        "address_type": 1,
                        "street": "100 west street",
                        "address_line_1": "apt no 7",
                        "address_line_2": "opposite to college park",
                        "city":"Dallas",
      /                                                    /
                                 "is_active":0
                          }

                  ]
}
```

2) **Retrieve client**

GET https://DomainName/api/v1/clients/<client Id>

Sample Request:

GET clients/<client id>

```
Content-Type: application/json
Response Body:
{
  "id": 27,
  "accountNo": "000000027",
  "status": {
              "id": 300,
              "code": "clientStatusType.active",
              "value": "Active"
  },
  "active": true,
  "activationDate": [
                        2013,
                        1,
                        1
```

```
"displayName": "savings test",
"officeId": 1,
"officeName": "Head Office",
"timeline": {
              "submittedOnDate": [
                                  2013,
                                  1,
                                  1
                                 ],
              "submittedByUsername": "mifos",
              "submittedByFirstname": "App",
              "submittedByLastname": "Administrator",
               "activatedOnDate": [
                                   2013,
                                   1,
                                   1
                                 ],
             "activatedByUsername": "mifos",
             "activatedByFirstname": "App",
             "activatedByLastname": "Administrator"
             },
"savingsProductId": 4,
"savingsProductName": "account overdraft",
"groups": [],
 "address":[
              {
                      "clientId":1,
                      "addressId":2,
                      "address_type": 1,
                      "street": "507 west street",
                      "address_line_1": "apt no 5",
                      "address_line_2": "opposite to college park",
                        "postal_code":"40041",
                        "is_active":1
                },
                {
                        "clientId":1,
                        "addressId":3,
                        "address_type": 1,
                        "street": "100 west street",
                        "address_line_1": "apt no 7",
                        "address_line_2": "opposite to college park",
                        "city":"Dallas",
                        "state_province_id":1,
                        "country_id":2,
                        "postal_code":"40041",
                        "is_active":0
                 }



        ]
}
```

3) **Update a client**

PUT https://DomainName/api/v1/clients/<clientId>

Sample Request

PUT clients/<clientid>

```
    /                                                    /

    "externalId":"786444000fm7",
    "Address line 1":"Apt 5"
}
```

No labels

4 Comments

Adi Raju

Can you please capture the need for this enhancement, what is considered as part of the requirements and what is left out (in other words limitations) at the beginning of this page. Scoped requirements can set proper context for the technical review.

Few questions that come to my mind based on above documentation:

1. Can the address API be generalised by linking it to entity than being specific to clients. API could be /{entityType}/{entityId}/addresses) This is followed currently in many of the APIs. Will also be extensible in future to capture address for group/office/center etc etc.
2. How is validation done on the different fields?

Reply • Jul 05, 2016

1. Can the address API be generalised by linking it to entity than being specific to clients. API could be /{entityType}/{entityId}/addresses) This is followed currently in many of the APIs. Will also be extensible in future to capture address for group/office/center etc etc.

From API readability, it is better to keep separate API for each of the entity ex: /client/<client_id>/addresses, office/<office_id>/address, but internal services and tables are reused except entity address mapping table

1. How is validation done on the different fields?

Validations are defined in the m_filed_configuration, that has ability to enable/disable the fields and can be marked a field as mandatory and also specify the regex for validation of each of the fields.

Reply • Jul 05, 2016

1. Use the API endpoint as addresses than address
2. Address type looks more of an attribute of address than entity, use parameterisation to get subset of addresses like /addresses?type="office"
3. Use POST API for changing status of address using commands on a specific address id like /addresses/{addressId}?command=active/inactive
4. It is possible that client will have multiple HOME or OFFICE addresses, but one of them might be identified as primary. Does this modelling useful or required from business point of view?
5. One of the address should be identified as communication address, is this something required from business?
6. Audit trail is maintained on m_address, but in cases when address status is changed, is there a need to manage the audit trail? In such cases is it required to capture the date detail as specific attribute or generic audit trail?
7. How do you plan to manage addresses that are same like home, office and communication address are same? How will edits work here, change in home address needn't change the office address.
8. What does entity, table and field mean in m_field_configuration?
9. I do not see any APIs to manage the m_field_configuration
10. How UI would know if they have to display a textbox, dropdown or a multi-select for a given attribute?
11. How do you plan to link code/code-value to any attribute in address table?
12. Please add template APIs for all the possible POST/PUT APIs
13. Sample request in "update a client" section is not right, you wouldn't know which address to update
14. How do you manage history of addresses? With what I can make out from above explanation, if I change the HOME address, I do not see the older address info maintained any where.

Reply • Jul 08, 2016

/ /

type= office

agree, type will be type id and the label

3. Use POST API for changing status of address using commands on a specific address id like /addresses/{addressId}?command=active/inactive

agree

4. It is possible that client will have multiple HOME or OFFICE addresses, but one of them might be identified as primary. Does this modelling useful or required from business point of view?

type of address will indirectly imply the primary or significance of the address

5.One of the address should be identified as communication address, is this something required from business?

that can be one of the address type or present address can be used as communication address, it is implicit and not planning to capture explicitly.

6.Audit trail is maintained on m_address, but in cases when address status is changed, is there a need to manage the audit trail? In such cases is it required to capture the date detail as specific attribute or generic audit trail?

will depend on generic audit logs

7.How do you plan to manage addresses that are same like home, office and communication address are same? How will edits work here, change in home address needn't change the office address.

it will notify to the user that change in this address will impact other addresses if she/he proceeds then all the other address will be impacted otherwise user has to create separate address for other type.

8. What does entity, table and field mean in m_field_configuration?

entity means client, groups, offices, user etc, to that entity address can be attached

table means m_address table, we are planning to rename it as enitity and mapped_to_enitity

9.I do not see any APIs to manage the m_field_configuration

Field level configuration will be implemented later, in first phase all the fields will be available.

10. How UI would know if they have to display a textbox, dropdown or a multi-select for a given attribute?

Similar to way how data table is doing

12.Please add template APIs for all the possible POST/PUT APIs

agree

13.Sample request in "update a client" section is not right, you wouldn't know which address to update

will add the address type

14.How do you manage history of addresses? With what I can make out from above explanation, if I change the HOME address, I do not see the older address info maintained any where.

yes, at present it out of scope, but any this can be extract from system audit logs

thanks for the detailed analysis and feedback

Reply • Jul 08, 2016