

```
{
  "Name" : "William Andy",
  "SenderRefID" : "@@SenderRefID@",
  "TaxID" : "123-45-6793"
}
```



## Solution 1: Create a Contact Integration View

1. Extend the `integration.schemas.trn.ta.contact-1.0.schema.json` file and add new fields.
  - a) Create a new package.
    - Right-click on **config.integration** folder and click **New → Package**.
    - Enter **schemas.si.ta** as the new package name.
  - b) Create a new schema file.
    - Right-click on **schemas.si.ta** folder and click **New → File**.
    - Enter **contact-1.0.schema.json** as the new file name.
  - c) Add schema header information and new definitions.

```
{
  "$schema" : "http://json-schema.org/draft-04/schema#",
  "x-gw-combine" : [
    "trn.ta.contact-1.0"
  ],
  "definitions" : {
    "ContactDetails" : {
      "type" : "object",
      "properties" : {
        "AssignedUser" : {
          "$ref" : "#/definitions/AssignedUser"
        },
        "BankAccounts" : {
          "type" : "array",
          "items" : {
            "$ref" : "#/definitions/BankAccountDetails"
          }
        }
      },
    },
    "MaritalStatus" : {
      "type" : "string",
      "x-gw-type": "typekey.MaritalStatus"
    },
    "SenderRefID" : {
      "type" : "string"
    },
    "TaxID" : {
      "type" : "string"
    }
  }
},
```

```

"AssignedUser" : {
  "type" : "object",
  "properties" : {
    "Name" : {
      "type" : "string"
    }
  }
},
"BankAccountDetails" : {
  "type" : "object",
  "properties" : {
    "BankName" : {
      "type" : "string"
    },
    "BankAccountType" : {
      "type" : "string",
      "x-gw-type" : "typekey.BankAccountType"
    }
  }
}
}
}
}
}

```

2. Create an Enhancement for the class **KeyableBean**. Add a property getter called **SenderRefIDPlaceholder\_Ext** which returns “**@@SenderRefID@@**”.

- a) Create a new package.
  - Right-click on **gsrsrc** folder and click **New → Package**.
  - Enter **si.ta.enhancements.messaging** as the new package name.
- b) Create a new Gosu class.
  - Right-click on **si.ta.enhancements.messaging** folder and click **New → Gosu Enhancement**.
  - Enter **RootPlaceholderEnhancement** as the as the new enhancement name.
  - Enter **KeyableBean** (entity) for enhancement type.
- c) Add property getter called **SenderRefIDPlaceholder\_Ext** which returns **@@SenderRefID@@** string.

```

package si.ta.enhancements.messaging

enhancement RootPlaceholderEnhancement : KeyableBean {
  property get SenderRefIDPlaceholder_Ext() : String {
    return "@@SenderRefID@"
  }
}

```

3. Extend the **integration.mappings.trn.ta.contact-1.0.mapping.json** file and add new fields. Map the **SenderRefID** field using **ABContact.SenderRefIDPlaceholder\_Ext**.

- a) Create a new package.
  - Right-click on **integration** folder and click **New → Package**.
  - Enter **mappings.si.ta** as the new package name.
- b) Create a new mapping file.
  - Right-click on **mappings.si.ta** folder and click **New → File**.
  - Enter **contact-1.0.mapping.json** as the as the new file name.
- c) Add mapping header information and new mappers.

```

{
  "schemaName" : "si.ta.contact-1.0",
  "combine" : [
    "trn.ta.contact-1.0"
  ],
  "mappers" : {
    "ContactDetails" : {
      "schemaDefinition" : "ContactDetails",
      "root" : "entity.ABContact",
      "properties" : {
        "AssignedUser" : {
          "path" : "ABContact.AssignedUser",
          "mapper" : "#/mappers/AssignedUser"
        },
        "BankAccounts" : {
          "path" : "ABContact.BankAccounts",
          "mapper" : "#/mappers/BankAccountDetails"
        },
        "MaritalStatus" : {
          "path": "(ABContact as ABPerson).MaritalStatus",
          "predicate": "ABContact typeis ABPerson"
        },
        "SenderRefID" : {
          "path" : "ABContact.SenderRefIDPlaceholder_Ext"
        },
        "TaxID" : {
          "path" : "ABContact.TaxID"
        }
      }
    },
    "AssignedUser" : {
      "schemaDefinition" : "AssignedUser",
      "root" : "entity.User",
      "properties" : {
        "Name" : {
          "path" : "User.DisplayName"
        }
      }
    },
    "BankAccountDetails" : {
      "schemaDefinition" : "BankAccountDetails",
      "root" : "entity.BankAccount",
      "properties" : {
        "BankName" : {
          "path" : "BankAccount.BankName"
        },
        "BankAccountType" : {
          "path" : "BankAccount.AccountType"
        }
      }
    }
  }
}

```

#### 4. Create the new filters.

- a) Create a new package.
  - Right-click on **integration** folder and click **New → Package**.
  - Enter **filters.si.ta** as the new package name.
- b) Create a new filter file.
  - Right-click on **filters.si.ta** folder and click **New → File**.
  - Enter **fraud\_check-1.0.gql** as the as the new file name.

- Select **Text** as new file type association.

Note: Ignore the **Plugins supporting \*.gql file found.** prompt. Click **Ignore extension**.

- c) Add filtered fields.

```
{  
  Name,  
  SenderRefID,  
  TaxID  
}
```

- d) Create a new filter file.

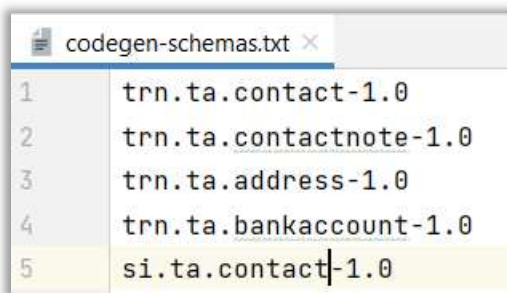
- Right-click on **filters.si.ta** folder and click **New → File**.
- Enter **contact\_api-1.0.gql** as the as the new file name.

- e) Add filtered fields.

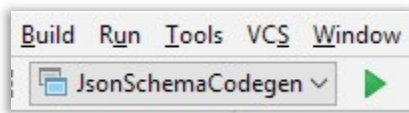
```
{  
  AssignedUser {  
    Name  
  },  
  BankAccounts {  
    BankAccountType,  
    BankName  
  },  
  Name,  
  PrimaryAddress {  
    AddressLine1,  
    AddressType,  
    City,  
    PostalCode,  
    State  
  },  
  TaxID  
}
```

## 5. Generate wrapper classes.

- a) Add the fully-qualified schema name, **si.ta.contact-1.0**, in the **codegen-schemas.txt** file.



- b) Run **JsonSchemaCodegen**.



```
Run JsonSchemaCodegen
C:\jdk1.8.0_152\bin\java -server -ea -Xdebug -Djava.awt.headless=true -Dgw.port=8880 -Xmx4g -Dgw.server.mode=dev -Dgw.debug=true -Dgw.webapp.dir=idea/webapp -
Looking for the devroot starting from C:\GW10\TrainingApp\
Detected CONFIG environment, root: C:\GW10\TrainingApp
Classpath modules: [configuration]
Module path: [configuration]
--- Json Schema Wrapper Code Generator Tool ---

--- Initializing ---

--- Initialization Finished ---

2018-08-06 20:11:00,843 INFO Generating jsonschema wrapper classes for all modules
2018-08-06 20:11:01,509 INFO ExternalConfigurationProviderPlugin is disabled. All configuration substitutions without default values will be left unchanged.
2018-08-06 20:11:01,732 INFO Parsing C:\GW10\TrainingApp\modules\configuration\config\config.xml for registry. No substitution is supported at that level.
2018-08-06 20:11:01,843 INFO Generating wrapper classes for schema demo.ta.user-1.0 in module C:\GW10\TrainingApp\modules\configuration
2018-08-06 20:11:01,878 INFO Generating wrapper classes for schema trn.ta.contactnote-1.0 in module C:\GW10\TrainingApp\modules\configuration
2018-08-06 20:11:01,882 INFO Generating wrapper classes for schema trn.ta.address-1.0 in module C:\GW10\TrainingApp\modules\configuration
2018-08-06 20:11:01,888 INFO Generating wrapper classes for schema si.ta.contact-1.0 in module C:\GW10\TrainingApp\modules\configuration
2018-08-06 20:11:01,906 INFO Generating wrapper classes for schema trn.ta.contact-1.0 in module C:\GW10\TrainingApp\modules\configuration

Process finished with exit code 0
```

## 6. Deploy code changes.

- From the **Studio** menu, **Restart the server**.

## 7. Perform verification steps.

- Write code that will generate JSON output for ABConstruction (PublicID = absample:3) using filter `contact_api`.

```
uses gw.api.database.Query
uses gw.api.database.Relop
uses gw.api.json.JsonConfigAccess
uses gw.api.json.mapping.JsonMappingOptions

// Query for Contact
var queryObj = Query.make(ABContact)
queryObj.compare(ABContact#PublicID, Relop.Equals, "absample:3")
var targetObj = queryObj.select().AtMostOneRow

// Create JsonMapper object
var jsonMapper = JsonConfigAccess.getMapper("si.ta.contact-1.0", "ContactDetails")

// Create JsonMapperOptions object
var mappingOpts = new JsonMappingOptions().withFilter("si.ta.contact_api-1.0")

// Create TransformResult object
var transformResult = jsonMapper.transformObject(targetObj, mappingOpts)

// Create output
var payloadJSON = transformResult.toPrettyJsonString()
print(payloadJSON)
```

- Write code that will generate JSON output for William Andy (PublicID = ab:5) using filter `contact_api`.

```
uses gw.api.database.Query
uses gw.api.database.Relop
uses gw.api.json.JsonConfigAccess
uses gw.api.json.mapping.JsonMappingOptions

// Query for Contact
```

```

var queryObj = Query.make(ABContact)
queryObj.compare(ABContact#PublicID, Relop.Equals, "ab:5")
var targetObj = queryObj.select().AtMostOneRow

// Create JsonMapper object
var jsonMapper = JsonConfigAccess.getMapper("si.ta.contact-1.0", "ContactDetails")

// Create JsonMapperOptions object
var mappingOpts = new JsonMappingOptions().withFilter("si.ta.contact_api-1.0")

// Create TransformResult object
var transformResult = jsonMapper.transformObject(targetObj, mappingOpts)

// Create output
var payloadJSON = transformResult.toPrettyJsonString()
print(payloadJSON)

```

c) Write code that will generate JSON output for William Andy (PublicID = ab:5) using filter fraud\_check.

```

uses gw.api.database.Query
uses gw.api.database.Relop
uses gw.api.json.JsonConfigAccess
uses gw.api.json.mapping.JsonMappingOptions

// Query for Contact
var queryObj = Query.make(ABContact)
queryObj.compare(ABContact#PublicID, Relop.Equals, "ab:5")
var targetObj = queryObj.select().AtMostOneRow

// Create JsonMapper object
var jsonMapper = JsonConfigAccess.getMapper("si.ta.contact-1.0", "ContactDetails")

// Create JsonMapperOptions object
var mappingOpts = new JsonMappingOptions().withFilter("si.ta.fraud_check-1.0")

// Create TransformResult object
var transformResult = jsonMapper.transformObject(targetObj, mappingOpts)

// Create output
var payloadJSON = transformResult.toPrettyJsonString()
print(payloadJSON)

```