Creación de Malla de Validación

El script de Malla de Validación, que valida y verifica los datos recolectados a través del aplicativo RIT, necesita de un diccionario, denominado "*malla*", de tipo JSON que le permita identificar las reglas de validación que debe aplicar en el conjunto de datos. Por medio del siguiente aplicativo, se darán los pasos que se deben seguir para crear el archivo JSON.

1. Contenido de la malla de validación

La malla contiene las distitas reglas que debe cumplir cada una de las variables almacenadas en el conjunto de datos, estas reglas son:

- Los posibles valores que puede tomar el registro en la variable determinada.
- La condición que habilita la respuesta de una variable, en el caso que esta dependa de una respuesta en una variable adicional.

Un ejemplo de las reglas que están contenidas en la malla de validación es el siguiente:

- La variable relacionada al número de teléfono del titular del hogar participante, solo puede ser una cadena con valores numéricos, cuya longitud debe ser de 10 carácteres y debe iniciar con un 3 o un 6.
- La variable relacionada con el número de teléfono adicional del titular del hogar debe ser únicamente respondida si la variable relacionada con la pregunta "Cuenta con un número de telefono adicional" haya sido respondida con el valor "SI".

A continuación, se detallará como se almacenan estas reglas en la malla para su respectiva validación a través del script de Python.

2. Estructura de la malla de validación

La malla de validación está contenida en un archivo de tipo JSON, este es un archivo de texto plano que contiene una colección de pares nombre, valor. Esto permite almacenar atributos sobre un objeto, en el caso del trabajo de validación, permite almacenar las distintas reglas que debe cumplir cada variable del conjunto de datos.

La estructura definida para la malla de validación es la siguiente

```
{"variable_1":{
    "condicion":,
    "valores":,
    "iand":,
    "opcional":,
    "excluida_PTA":
},
"variable_2":{
    ...
},
}
```

Donde cada atributo definido va a almacenar las reglas de validación sobre la variable que se va a evaluar. A continuación se explica de que trata cada uno de los atributos definidos.

3. Condicion

En el atributo de condición se debe almacenar aquellas respuestas que debe tener la variable que condiciona a la variable que se está evaluando, el objeto que se almacena dentro de este atriburo será el nombre de la variable que condiciona como la respuesta que debe tomar para habilitar la variable a evaluar.

```
"condicion":{
    "variable_condiciona":[respuesta_habilitadora_1, ..., respuesta_habilitadora_j]
},
```

Continuando con el ejemplo de la variable relacionada con la respuesta de número de teléfono adicional, la estructura del atributo condición será:

```
"variable":{
"condicion":{
    "telefono_adicional":["SI"]
    },
...
},
```

3.1 Anotaciones sobre la condición de la variable:

En el caso que la variable a evaluar no dependa de la respuesta de una variable adicional, entonces el atributo de la condición será nulo, para esté caso en el archivo JSON su estructura será:

```
"condicion":null,
```

En el caso que la variable a evaluar depende de la respuesta en dos o más variables, la estructura del atriburo condición será el siguiente:

```
"condicion":{
    "variable_condiciona_1":[respuesta_habilitadora_1, ..., respuesta_habilitadora_j],
    "variable_condiciona_2":[respuesta_habilitadora_1, ..., respuesta_habilitadora_n]
},
```

4. Valores

El atributo Valores, contiene los posibles valores que puede tomar la respuesta en la variable a evaluar. La estrucura dispuesta para este atributo es la siguiente:

```
"valores":{
    "valor":,
    "Tipo":
},
```

Los valores dentro de cada atributo están determinados según el tipo de valor que puede tomar la variable.

4.1 Tipos de Valor

Las respuestas contenidas dentro de la variable pueden ser de cinco tipos distintos: numérica, cadena de texto, expresión regular, lista de valores y lista de lista de valores.

4.1.1 Valor de tipo numérico

Se refiere a los casos en los que los posibles valores que puede tomar una variable son valores únicamente numéricos. Para este caso la estructura del atributo será la siguiente:

```
"valores":{
    "valor":[valor_1, ..., valor_n],
    "Tipo":"int"
},
```

Ejemplo

```
"variable":{
    "condicion":...,
    "valores"{
        "valor":[1,2,3],
        "Tipo":"int"
    },
    ...
}
```

4.1.2 Valor de tipo cadena de texto

Cómo lo indica su nombre, se refiere a los casos donde los posibles valores que puede tomar una variable son cadenas de texto predetermianadas. Su estructura será la siguiente:

```
"valores":{
    "valor":[valor_1, ..., valor_n],
    "Tipo":"str"
},
```

Ejemplo

```
"variable":{
    "condicion":...,
    "valores"{
        "valor":["SI","NO"],
        "Tipo":"str"
    },
    ...
}
```

4.1.3 Valor de tipo Expresión Regular

En el caso que el valor de respuesta de la variable no esté establecido por una lista predeterminada de opciones pero contiene condiciones sobre la respuesta, por ejemplo, el número de teléfono o el correo electrónico, se aplica una

validación de los valores a través de una Expresión Regular, que es una cadena de texto que permite identificar coincidencias de caracteres en cadenas de texto. Su estructura en el atributo de valores será el siguiente:

```
"valores":{
    "valor":expresion_regular,
    "Tipo":"regex"
},
```

Ejemplo Para el caso del número de teléfono cuya longitud debe de ser de 10 carácteres numéricos y debe iniciar con el número 3 o 6, la expresión regular estará definida cómo: `"^[36]\\d{9}$"`. En el archivo JSON estará reflejada cómo:

```
"variable":{
    "condicion":...,
    "valores"{
        "valor":"^[36]\\d{9}$",
        "Tipo":"regex"
    },
    ...
}
```

4.1.4 Valor de tipo Lista

En el caso que la respuesta en la variable a evaluar sea de opción multiple, es decir, puede seleccionar una o más respuestas, el proceso de validación verificará que cada respuesta almacenada en la variable esté dentro de la lista de posibles valores que se pueden seleccionar en la variable. Su estructura será la siguiente:

```
"valores":{
    "valor":[valor_1, ..., valor_n],
    "Tipo":"list"
},
```

Ejemplo

```
"variable":{
    "condicion":...,
    "valores"{
        "valor":[1,2,3,4,5],
        "Tipo":"list"
    },
    ...
}
```

4.1.5 Valor de tipo lista de listas

En el caso que la respuesta de una variable sea de opción multiple sobre multiples objetos o respuestas anteriores, el valor que se almacena en la variable será una lista que contiene las respuestas multiples para cada objeto o respuesta seleccionada. En esto caso, el proceso de validación será verificar en cada lista que los valores almacenados dentro se

encuentren dentro de la lista de opciones determinado. Su estructura en el archivo JSON será igual a la estructura de tipo lista, con la diferencia que para este caso el atributo `Tipo` será `listlist`.

4.2 Anotaciones sobre el atributo Valor

En el caso que la variable pueda tomar cualquier tipo de valor, al igual que con el atributo `condicion`, el atributo `Valores` será nulo, por lo que su estructura será:

```
"variable":{
    "condicion":...,
    "valores": null,
    ...
}
```

5. iand

El atributo `iand`, es un valor lógico, es decir, `true` o `false`, que determina lo siguiente:

> En el caso que la condición para la variable contenga 2 o más variables, el atributo `iand` determina, si se deben cumplir todas las condiciones o se debe cumplir al menos una de las condiciones.

Con esto en cuenta, el valor del atributo `iand` será `true`, únicamente en el caso que se deban cumplir todas las condiciones de respuesta para la variable a evaluar, en el caso que solo se debe cumplir al menos una condición o no contiene más de dos variables en la condición, el valor del atributo `iand` será `false`.

```
"variable":{
    "condicion":...,
    "valores":...,
    "iand": false,
    ...
},
```

6. Opcional

Al igual que el atributo `iand`, el valor de opcional será un valor lógico, y determina si la variable que está siendo evaluada corresponde a una pregunta cuya respuesta es obligatoria u opcional. Un ejemplo de esto, es la pregunta de Segundo nombre, es una pregunta que es opcional dado que no todas las personas tienen un segundo nombre. Con esto encuenta, el atributo `opcional` será `true` en el caso que la variable corresponda a una pregunta opcional, en cualquier otro caso su valor será `false`.

```
"variable":{
    "condicion":...,
    "valores":...,
    "iand": ...,
    "opcional": false,
    ...
},
```

7. excluida_PTA

Es un atributo de valor lógico, y determina la siguiente condición:

> La variable a evaluar se refiere a una pregunta que debe ser contestada sin importar el resultado de las variables relacionadas con las preguntas, *"Desea Participar en el Programa"*, *"Dispone de Agua"* y *"Dispone de Tierra"*.

Dado así, en el caso que la variable no dependa de la respuesta de estas preguntas en el conjunto de datos, el valor del atributo será `true`, dado el caso que si dependa, su valor será `false`.

```
"variable":{
    "condicion":...,
    "valores":...,
    "iand": ...,
    "opcional":...,
    "excluida_PTA": true
},
```

8. Estructura de la malla por medio de un ejemplo

Si se supone que se tiene la variable `pregunta_1` con las siguientes características:

- Depende que la respuesta de la variable `ZONA` sea 1 ó 2.
- La respuesta de la pregunta solo puede ser SI, NO o NO SABE.
- Es una pregunta obligatoria y que está excluida de las respuestas Participar, Tierra y Agua.

Con esto en cuenta, la estructura de las reglas de validación para esta variable será:

```
{
    "variable_n-1":...,
    "pregunta_1":{
        "condicion":{
            "ZONA":[1,2]
        },
        "valores":{
            "valor":["SI","NO","NO SABE"],
            "Tipo":"str"
        },
        "iand": false,
        "opcional": false,
        "excluida_PTA": true
    },
    "variable_n+1":...,
}
```

9. Formato excel para la malla de validación

Para facilitar el proceso de definición y estructuración de la malla de validación como un archivo de tipo JSON, se ha dispuesto generado un script en Python capaz de crear el archivo de tipo JSON a partir de un archivo excel con una estructura definida.

Este archivo excel va a contener las distintas reglas que debe cumplir cada variable, una vez se ha definido de forma correcta, a través de un script de Python se generará la malla de validación para la respectiva validación del conjunto de datos dispuesto.

9.1 Estructura del formato excel

La estructura que se ha dispuesto para la creación de la malla de validación desde el archivo excel está dividido en dos hojas.

9.1.1 Hoja Validaciones

La hoja validaciones del archivo excel contiene toda la información relacionada con la validación que se le debe realizar respecto a las condiciones que habilitan la respuesta de la variable. El nombre de la hoja es `Validaciones` y contiene una tabla con la siguiente estructura:

- **variable**: Contiene el nombre de la variable a evaluar
- **dependiente**: Contiene el nombre de la variable sobre la que depende la variable a evaluar para habilitar su respuesta. En el caso que haya más de una condición, se debe crear un nuevo registro con el mismo nombre de la variable y el nombre de la columna de la que depende. Además, si la variable no depende de ninguna variable adicional, el valor en la columna debe estar en blanco.
- **condicion**: Contiene la respuesta específica que se debe responder en la variable sobre la que se depende para habilitar la respuesta de la variable a evaluar. En el caso que la variable pueda tomar dos o más respuestas, estas deben ir separadas por el carácter `|` y sin espacio entre las respuestas.
- **tipo_validacion**: Columna que indica si el valor que debe tomar la columna dependiente es de tipo numérico o es una cadena de texto. En el caso que el valor sea numérico, el valor de la columna será `int`. En el caso contrario se deberá dejar vacía la columna.
- **iand**: Columna que indica si para las condiciones que habilitan la respuesta de la pregunta, se deben cumplir todas o se debe cumplir al menos una de esta. En el caso que se deban cumplir todas, el valor en la columna debe ser `1` en el primer registro de la variable, en el caso que no hayan dos o más condiciones o se debe cumplir al menos una, se debe dejar vacía la columna
- **excluye_PTA**: El valor indica si la variable se debe excluir de la condición de las preguntas, Participar, Tierra y Agua, en el caso que esté exluida, su valor será `SI`, caso contrario se debe dejar vacío.
- **variable_opcional**: El valor de la columna indica si la respuesta dentro de la variable es opcional u obligatoria. En el caso que la respuesta de la variable sea opcional, el valor será `SI`, caso contrario, se debe dejar vacio el valor en la columna.

A continuación, se presentan varios ejemplos para distintos casos de validación:

Variable sin ningun tipo de condición y excluida

variable	dependiente	condicion	tipo_validacion	iand	excluye_pta	variable_opcional
nombre_variable					SI	

Variable con una única condición de tipo numérico, es opcional y no exluida

variable	dependiente	condicion	tipo_validacion	iand	excluye_pta	variable_opcional
nombre_variable	variable_condicion	1|2|3	int			SI

Variable con dos condiciones, una numérica y otra con cadenas de texto, se deben cumplir todas las condiciones, es excluida y no es opcional

variable	dependiente	condicion	tipo_validacion	iand	excluye_pta	variable_opcional
nombre_variable	variable_condicion_1	valor1|valor2		1	SI	
nombre_variable	variable_condicion_2	5	int		SI	

En esta tabla se deben almacenar todas las condiciones expuestas para cada variable, en el caso que la variable no tenga ninguna condición se debe añadir de igual el registro para la variable ya que esto permitirá identificar las variables que deben de existir en el conjunto de datos a validar.

9.1.2 Hoja Valores

La hoja valores, cómo su nombre lo índica, contiene la información relacionada con los valores que puede tomar cada variable, el nombre de la hoja es `Valores` y almacena una tabla con las siguientes columnas:

- **variable**: El nombre de la variable que se va a evaluar.
- **valores**: Los posibles valores que puede tomar la variable, en el caso que la variable pueda tomar multiples valores, estos deben ir separados por el caracter `|`, y en el caso que sea una expresión regular, el valor en la columna solo debe ser la expresión.
- **tipo_valor**: Se refiere al tipo de valor que puede tomar, el valor que se debe colocar puede ser `int, str, regex, list, listlist` según el tipo de valor explicado en la sección 4.1.

A continuación se da un ejemplo para cada tipo de valor:

variable	valores	tipo_valor
variable_numerica	1|2|3	int
variable_texto	SI|NO|NO SABE	str
variable_regex	^[36]\d{9}$	regex
variable_list	12|13|14	list
variable_listlist	valor1|valor2|valor3	listlist

En el caso que la variable no deba ser evaluada respecto a los valores que toma, esta no debe ser añadida en esta tabla de valores.

9.2 Ruta de almacenamiento del archivo de tipo excel

Para poder hacer una trazabilidad y mantener un correcto almacenamiento de los archivos utilizados durante la malla de validación, el archivo con la malla de validación en excel debe de ser guardado en la carpeta `xlsx`, que está almacenada dentro de la carpeta `data`. Además el nombre de el archivo debe identificar el número identificado de la encuesta sobre la que se va a realizar la validación con esa malla.

```
|- Carpeta del Proyecto
    |- data
        |- json
        |- xlsx
            |- Aquí debe guardar el archivo excel
    ...
```

10. Creación de la malla de validación de tipo JSON

Si la creación de la malla de validación fue realizada a través del archivo excel, el proceso de exportación a la malla de tipo JSON, se puede realizar a través de un Script de Python dispuesto para generar este archivo. A continuación se van a repasar los requerimientos para su correcto uso como los pasos a seguir.

==Es necesario que los nombres de las hojas del archivo excel tengan el nombre dispuesto para cada una de estas, en el caso que no este correcto su nombre el Script de Python no podrá ser ejecutado ya que no identificara cada hoja del

archivo.==

10.1 Instalar Python y las librerias requeridas

Para poder ejecutar el script de Python para la creación de la malla de validación así como cualquier otro código desarrollado en el lenguaje Python, para realizar este proceso puede seguir el siguiente enlace y seguir los pasos 1 y 2 de la seccion *Para instalar Python 3.6 y pip (Windows)*.

Una vez ha instalado el lenguaje Python en su computadora, es necesario instalar las librerias necesarias para el correcto funcionamiento del Script. Estas librerias se encuentran almacenadas en el archivo *requeriments.txt* que se encuentra dentro de la carpeta del proyecto. Para realizar la instalación de las librerias realice el siguiente procedimiento.

1. A través del Explorador de archivos ubique la carpeta del proyecto e ingrese a esta.
2. Utilice el comando `shift + clic izquierdo` para abrir la ventana de opciones.
3. En la ventana de opciones seleccione la opción `Abrir en terminal`.
4. Dentro del terminal o consola de Windows, ingrese el siguiente comando: `pip install -r requeriments.txt`
5. Espera a que el proceso de instalación de las librerias termine.
6. Certifique que las librerias establecidas en el archivo requeriments fueron instaladas con el comando `pip list`, este comando le indicará las librerias de Python que están instaladas dentro de su terminal.
7. Cierre la consola o terminal de Windows.

Con la instalación del languaje Python, así como sus librerías, junto al archivo malla de validación debidamente diligenciado se puede proceder a generar el archivo de tipo JSON.

10.2 Ejecución del Script de Python para crear la malla de validación

Para la ejecución del Script que exporta el archivo excel con el formato JSON, siga los siguientes pasos:

1. Diríjase a la carpeta del proyecto.
2. Ubique el archivo `create_malla.py`
3. Haga clic derecho sobre el archivo para desplegar la ventana de opciones.
4. Diríjase a la opción `Abrir con`.
5. Seleccione la opción `Python`.

Este proceso desplegará una ventana de comandos en donde debe realizar el siguiente proceso:

1. Diligenciar el número identificador de la encuesta sobre la que se va a realizar la validación.
2. Seleccionar el archivo Excel que contiene la malla de validación.
3. Seleccionar la carpeta donde se va a exportar la malla de validación como un archivo de tipo JSON. El lugar dispuesto para almacenar este archivo es la carpeta `json` que está dentro de la carpeta `data` dentro de la carpeta del proyecto.

Una vez realice estos pasos, la malla de validación habrá sido exportada como un archivo de tipo JSON y su nombre estará indicado con el número identificador de la encuesta.

==El Script de Python contiene la utilidad que permite seleccionar la carpeta donde se desea almacenar la malla de validación exportada, sin embargo, para realizar el proceso de validación es necesario que el archivo se encuentre ubicado dentro de la carpeta `json` de lo contrario, el proceso de validación no podrá encontrar el archivo sobre el que realizará la validación.==