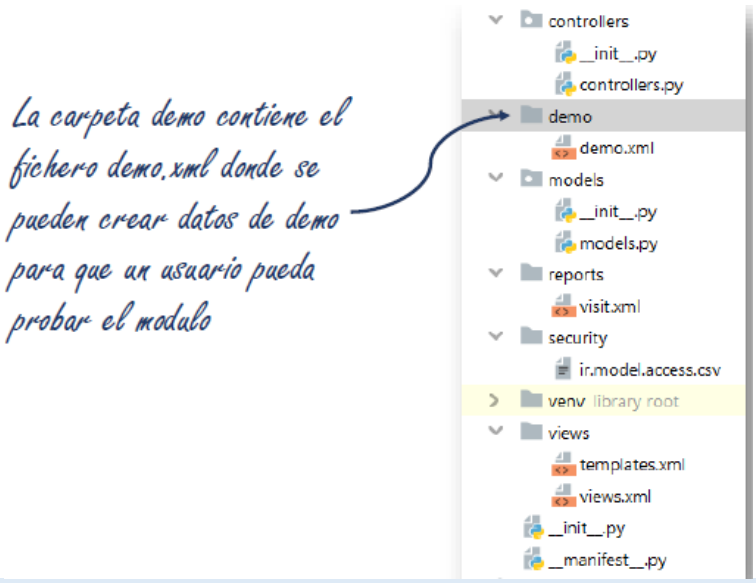
PRACTICA 8
OBJETIVO: DATOS DEMO
DESCRIPCIÓN:

Al ejecutar el comando odoo con el parámetro scaffold se crean los ficheros y carpetas que constituyen la forma básica de un modulo de Odoo

Entre las carpetas que se crean de manera automática está la carpeta demo que almacena ficheros xml con datos de demostración para poder probar un modulo

Dentro de la carpeta demo se crea también de manera automática el fichero demo xml para contener los datos de demostración

Cuando se instala un modulo el usuario tiene la opción de instalar estos datos de demo Los datos demo pueden son de utilidad para el desarrollador del modulo, con el fin de poder programarlo y depurarlo, ya que parte de datos precargados para hacer pruebas cada vez que necesita reinstalar el modulo



PASO 1: DATOS DEMO

El fichero demo xml tiene una estructura, parecida a las ya vistas, en la que aparecen varias etiquetas record record que representan registros que se van a almacenar en la base de datos

```
<odoo>
        <data>
                <record id="" model="">
                        <field name ="" > </field>
                        ...
                </record>
                ...
        </data>
</odoo>
```

Los atributos de las etiquetas record record son el modelo, model (la tabla de base de datos que va a contener el registro) y un id único para el registro contienen etiquetas field name con el nombre del campo, name y el valor que este recibe

Como es evidente, el nombre del campo, name tiene que hacer referencia a los atributos del modelo, esto es, a los campos definidos en el modelo a la derecha se representa un ejemplo del contenido del fichero demo xml

Para que los registros de demostración estén disponibles cada vez que se actualiza el modulo se debe incluir la ruta al fichero demo xml en lista data del fichero __manifest__.py ubicado en la carpeta que contiene el módulo de Odoo

PASO 2: DATOS DEMO

```
<odoo>
        <data noupdate ="1">
                <record id="" model="">
                        <field name=""> </field>
                        ...
                </record>
                ...
        </data>
</odoo>
```

En caso de que no se desee que se carguen los datos al actualizar el modulo, solo se quiere que se carguen cuando se instala, se puede añadir un atributo **noupdate** con valor 1 a la etiqueta

Esto hará que los datos de demostración no se carguen en la base de datos al actualizar el modulo
El contenido de un campo se puede proporcionar mediante la etiqueta <field> a través de un archivo externo utilizando el atributo file o mediante el cuerpo de la propia etiqueta
A continuación se enumeran los diferentes atributos que se pueden utilizar con la etiqueta Field:

a) Atributo name

name el nombre del campo.

b) Atributo **type**

type se utiliza para convertir el contenido del campo a un formato determinado. Los tipos disponibles son:

- **xml , html** extrae los valores del campo de las etiquetas field contenidas en el documento, para utilizar ids externos hay que ponerlos en la forma %(external_id). En el caso de necesitar usar el símbolo tanto por ciento % se deben poner dos seguidos %%.
- **file** el contenido del campo es una ruta a un archivo, y el valor del campo se guarda como module, ruta
- **char** se utiliza el contenido del campo directamente como el valor del campo, no realiza transformación alguna.
- **base64** utiliza el contenido del campo en codificación base64, se suele utilizar en combinación con el atributo file para cargar, por ejemplo, datos de imagen desde archivos.
- **int** convierte el contenido del campo en un número entero y lo utiliza como el valor del campo.
- ***float*** convierte el contenido del campo en un número en coma flotante y lo utiliza como el valor del campo.
- ***list***, ***tuple*** debe contener el mismo número de elementos con las mismas propiedades que los valores definidos en el campo, cada elemento se resuelve en un elemento de una tupla o lista generada, y la colección generada se establece como el valor del campo.

c) Atributo eval

eval evalúa su contenido como si fuera código Python. Esto permite definir valores que no son cadenas, ya que, normalmente, el contenido dentro de las etiquetas *<field>*siempre se evalúa como cadenas de caracteres.

Ejemplo: Esto se evalúa como la cadena '2.3' y no como float2.3

```
<field name="peso">2.3</field>
```

Ejemplo: Esto se evalúa como la cadena *'False*' y no al booleano *False*.

```
<field name="valor">False</field>
```

Si desea evaluar el valor de un flotante, un booleano u otro tipo, excepto una cadena, debe usar el atributo *eval*:

```
<fieldname="peso" eval="2.3" />
```

```
<field name="valor" eval="False" />
```

d) Atributo ref

ref sirve para especifica una *id externo de otro registro*. Es utilizado para establecer relaciones (*One2many*, *Many2many*…) entre los registros.

Ejemplo: El campo *company_id* es una relación de varios a uno (*Many2one*) entre el objeto de usuario y el objeto empresa, y *main_company* es el *id* asociado.

```
<field name="company_id" ref="main_company" />
```

Para asignar valores a campos de tipo relaciones uno a muchos (*One2many*) y muchos a muchos (*Many2many*) se utiliza una sintaxis especial.
Básicamente, para modificar una relación many2many se usa una tupla de tres elementos:

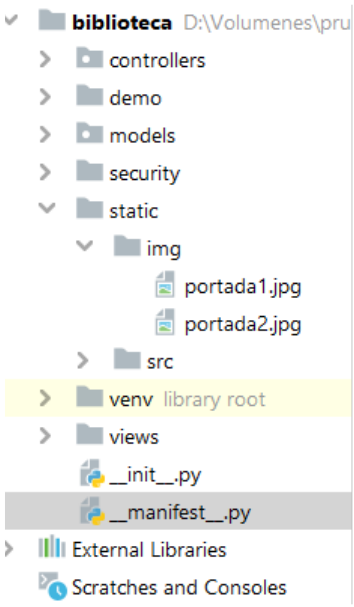
1. El primer elemento de la tupla es un comando numérico
2. Los otros dos elementos son valores que dependen del comando.

c.1) Hay seis comandos numéricos:

(0, _, dicc) agrega un nuevo registro creado a partir de los valores del diccionario dicc.
(1, id, dicc) actualiza un registro existente a partir del id proporcionado con los valores del diccionario dicc. No se puede utilizar con la función create().
(2, id, _) elimina el registro a partir del id proporcionado, lo borra de la base de datos. No se puede utilizar con la función create().
(3, id, _) elimina de la relación el registro a partir del id proporcionado, pero no elimina el registro en la tabla en la que este definido. No se puede utilizar en relaciones One2many. No se puede utilizar con la función create().
(4, id, _) agrega un registro existente de identificación id al conjunto. No se puede utilizar en relaciones One2many.
(5, _, _) elimina todos los registros del conjunto, lo que equivale a utilizar el comando 3 con cada registro de forma explícita. No se puede utilizar en relaciones One2many. No se puede utilizar con la función create().
(6, _, ids) reemplaza todos los registros existentes en el conjunto por la lista de ids proporcionada, equivale a usar el comando 5 seguido de un comando 4 para cada uno de los id en la lista ids. No se puede utilizar en relaciones One2many.
Los valores marcados como _ en la lista anterior se ignoran y pueden ser cualquier cosa, generalmente **0** o **False**.

PASO 3: EJEMPLO

Continuando con el ejemplo de los temas anteriores, vamos a añadir a nuestro modelo biblioteca libro dos registros que se precarguen cuando actualizamos el módulo
Vamos a crear primero un directorio llamado static dentro del modelo, esta carpeta suele contener los archivos estáticos adicionales al modulo, como son archivos CSS JS imágenes
En esta carpeta vamos a ubicar los ficheros de imagen que se utilizan como portadas de los libros
Así que creamos otra carpeta dentro de esta llamado img
Y ubicamos dos ficheros JPG para las portadas portada 1 jpg y portada 2 jpg



a) Contenido del fichero demo/demo.xml queda de la siguiente forma:

```xml
<odoo>
    <data>
        <record id="Object0" model="biblioteca.libro">
            <field name="name">El Señor de los Anillos</field>
            <field name="imagen" type="base64" file="biblioteca/static/img/portada1.jpg"/>
            <field name="pages">568</field>
            <field name="discontinued">False</field>
            <field name="language">Es</field>
            <field name="author_ids" eval="[(4,3,0)]"/>
        </record>


        <record id="Object1" model="biblioteca.libro">
            <field name="name">El Principito</field>
            <field name="imagen" type="base64" file="biblioteca/static/img/portada2.jpg"/>
            <field name="pages">33</field>
            <field name="discontinued">True</field>
            <field name="language">Fr</field>
            <field name="date_release">2020-10-15</field>
            <field name="author_ids" eval="[(4, ref('base.partner_admin'),0)]" />
            <field name="summary">El principito es una narración corta del escritor francés Antoine de Saint-Exupéry
            </field>
        </record>
    </data>
</odoo>
```

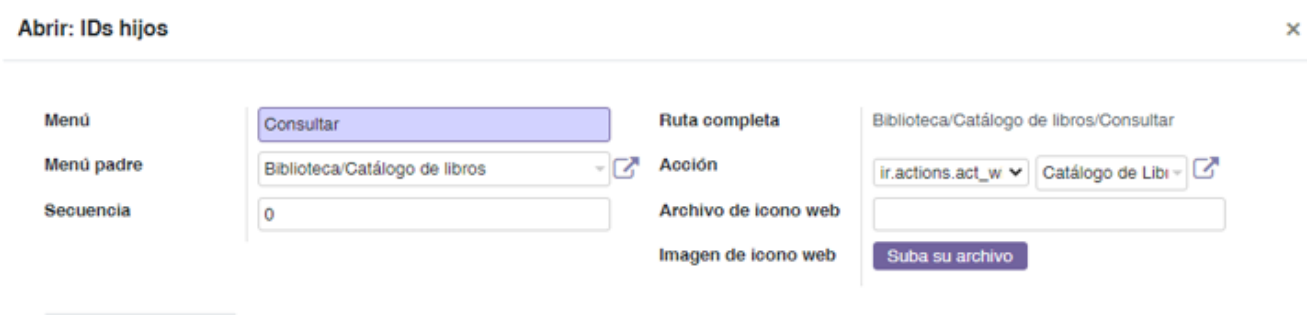
b) Modificar el contenido del fichero **__manifest__.py** ubicado en la carpeta que contiene el módulo de Odoo

Añadimos a la lista data el elemento 'demo/demo xml' que es la ruta al fichero que define donde se encuentran los datos de demostración, quedando de la siguiente forma:

```
'data': [
        'security/grupos.xml',
        'security/ir.model.access.csv',
        'views/vistas.xml',
        'views/views.xml',
        'views/templates.xml',
        'demo/demo.xml'
],
```

Para ver que se ha realizado correctamente, actualizamos el módulo biblioteca de Odoo y comprobamos que se han cargado los registros

PASO 4: Ejemplo



```
<odoo>
  <data>
    <record id="Autores0" model="biblioteca.autores">
      <field name="name">     Miguel de Cervantes Saavedra</field>
    </record>

    <record id="Libro0" model="biblioteca.libro">
      <field name="name">El Señor de los Anillos</field>
      <field name="imagen" type="base64" file="biblioteca/static/img/portada1.jpg"/>
      <field name="pages">568</field>
      <field name="discontinued">False</field>
      <field name="language">Es</field>
<!--         <field name="author_ids" eval="[(4,3,0)]"/>-->
    </record>


    <record id="Libro1" model="biblioteca.libro">
      <field name="name">El Principito</field>
      <field name="imagen" type="base64" file="biblioteca/static/img/portada2.jpg"/>
      <field name="pages">33</field>
      <field name="discontinued">True</field>
      <field name="language">Fr</field>
      <field name="date_release">2020-10-15</field>
<!--         <field name="author_ids" eval="[(4, ref('base.partner_admin'),0)]" />-->
      <field name="summary">El principito es una narración corta del escritor francés Antoine de Saint-Exupéry, que trata de la historia de un pequeño príncipe que parte de su asteroide a una travesía por el universo, en la cual descubre la extraña forma en que los adultos ven la vida y comprende el valor del amor y la amistad.
      </field>
    </record>

    <record id="Libro2" model="biblioteca.libro">
      <field name="name">El ingenioso hidalgo don Quijote de la Mancha</field>
      <field name="imagen" type="base64" file="biblioteca/static/img/portada3.jpg"/>
      <field name="pages">704</field>
      <field name="discontinued">False</field>
      <field name="language">Es</field>
      <field name="date_release">1605-1-1</field>
      <field name="author_ids" eval="[(4, ref('Autores0'),0)]" />
      <field name="summary">Don Quijote, hombre valiente y de lo más aventurero, se embarca (arrastrando a Sancho) en locas aventuras, hazañas, batallas y misiones, todas ellas en nombre de la justicia, el amor, y en definitiva, con el fin de salvar a cualquiera  que esté en problemas.
      </field>
    </record>
  </data>
</odoo>
```