Structure de la base de données

Modèle

Le menu Configuration > Gestion avancée administration > Modèle référence les ir.model, c'est-à-dire l'ensemble des modèles, objets, et tables présentes dans la base de données d'Open-Prod. L'ensemble des tables de l'ERP sont référencées, et plus notamment les ventes (sale.order), les lignes de ventes (sale.order.line), ...

De nombreuses créations d'enregistrements tels que le RSE, les workflows, ou encore les modèles de mails demandent de désigner un modèle (objet) sur lequel travailler. Tous les champs Modèle sont en relation avec cette table des ir.model.

Formulaire

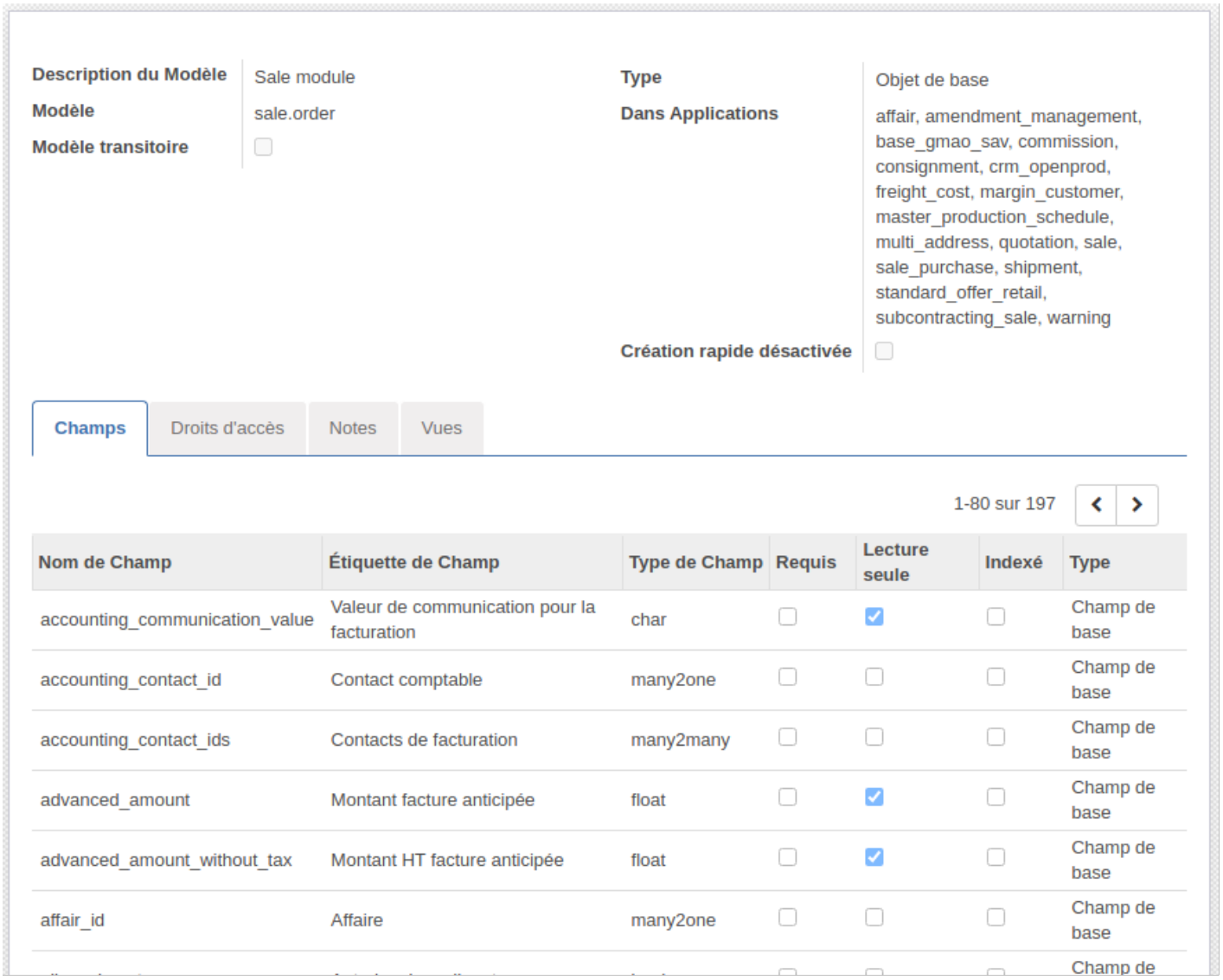
En-tête

- Description du modèle : nom commun de l'objet, affiché lors des recherches depuis les autres objets.
- Modèle : nom technique de l'objet, utilisé notamment dans les formules et les champs techniques.
- Modèle transitoire : la case est cochée si l'objet est une action ou une vue de transition comme un assistant de saisie. Vous ne pourrez pas afficher une liste ou un formulaire des modèles transitoires car ces derniers ne sont pas stockés. Ils sont utilisés pour renseigner un modèle non transitoire.
- Type : un modèle est un objet de base s'il est défini dans le code. Dans le cas contraire, il s'agit d'un objet personnalisé.
- Création rapide désactivée : si ce champ est coché, il ne sera pas possible de créer un nouvel enregistrement depuis sa sélection dans un autre objet (avec la liste déroulante et l'option « créer et modifier » ou en sauvegardant un nouvel enregistrement). Par défaut les objets concernés sont : les udm, les calendriers, les modèles de calendrier, les ressources, les catégories de ressources, les emplacements, les périodes, les exercices, les journaux, les taxes et les incoterms.
- Le champ Dans Applications indique les applications installées qui dépendent de l'objet. Ci-contre, on voit que de nombreux modules dépendent de l'installation de sale.order, notamment le module Commissions.

Onglet « Champs »

Cet onglet liste tous les champs appartenant à l'objet. À la création manuelle d'un objet, 7 champs sont créés automatiquement :

- Create_date : date de création de chaque enregistrement dans l'objet ;
- Create_uid : utilisateur qui a créé l'enregistrement ;
- Display_name : nom de l'enregistrement dans l'interface ;
- Id : numéro unique d’identification des enregistrements ;
- __last_update : date de dernière modification, qui s'actualise à chaque changement sur l'enregistrement ;
- Write_date : date de dernière modification, qui s'actualise à chaque changement sur l'enregistrement ;
- Write_uid : dernier utilisateur à avoir modifié l'enregistrement.



Description du Modèle	Sale module	**Type**	Objet de base
Modèle	sale.order	**Dans Applications**	affair, amendment_management, base_gmao_sav, commission, consignment, crm_openprod, freight_cost, margin_customer, master_production_schedule, multi_address, quotation, sale, sale_purchase, shipment, standard_offer_retail, subcontracting_sale, warning
Modèle transitoire	☐	**Création rapide désactivée**	☐

Champs | Droits d'accès | Notes | Vues

1-80 sur 197

Nom de Champ	Étiquette de Champ	Type de Champ	Requis	Lecture seule	Indexé	Type
accounting_communication_value	Valeur de communication pour la facturation	char	☐	☑	☐	Champ de base
accounting_contact_id	Contact comptable	many2one	☐	☐	☐	Champ de base
accounting_contact_ids	Contacts de facturation	many2many	☐	☐	☐	Champ de base
advanced_amount	Montant facture anticipée	float	☐	☑	☐	Champ de base
advanced_amount_without_tax	Montant HT facture anticipée	float	☐	☑	☐	Champ de base
affair_id	Affaire	many2one	☐	☐	☐	Champ de base
						Champ de

Une image contenant tableDescription générée automatiquement **Onglet « Droits d'accès »**

Cet onglet liste les groupes et droits associés attachés à l'objet. La liste est liée au contrôles d'accès. On y retrouve les différentes actions possibles ainsi que le groupe auquel ces droits sont rattachés.

Dans l'illustration, on peut voir qu'appartenir au groupe des utilisateurs des ventes permet uniquement d'avoir accès en lecture aux taxes dans les devis. Pour pouvoir les modifier, il faut au minimum faire partie des fonctionnels des ventes.

Onglet « Vues »

Cet onglet liste les vues associées à l'objet. La liste est liée aux vues présentes dans l'ERP. On y retrouve les vues de base définies par le code source, mais aussi les vues personnalisées définies depuis l'interface.

Une image contenant texteDescription générée automatiquement

Les champs

Le menu Configuration > Gestion avancée administration > Champs liste l'ensemble des champs présents dans le système ainsi que l'objet ou modèle associé.

Il est utile uniquement dans le cas de l'ajout de champs dans l'interface via le module de modification de vue. Il permettra de modifier des paramètres du champ créé à posteriori, comme par exemple les propriétés avancées et notamment le code Python qui calcule la valeur du champ créé.

Le formulaire

L'en-tête du formulaire des champs contient les champs suivants :

- Le nom du champ, visible en mode développeur en survolant le champ, et utilisé dans tous les champs techniques, notamment les formules en langage Python. Les « champs personnalisés », c'est-à-dire ceux créés manuellement, sont facilement identifiables car ils commencent tous par « x_ » ;
- Le modèle (objet) auquel est lié le champ ;
- L'étiquette de champ, c'est-à-dire le libellé dans l'interface ;
- Le champ d'aide, qui s'affiche en survolant un champ dans l'interface et contient des informations d'utilisation ;
- Le type de champ.

L'onglet « Propriétés de base »

Cet onglet permet de définir les paramètres de base du champ :

- S'il est traduisible, la valeur que contient le champ pourra être saisie dans plusieurs langues. Seuls les champs de type char et text sont traduisibles ;
- S'il est requis, le champ devra être saisi pour valider le formulaire dans lequel il apparaît.

Ne modifier cette option que pour les champs créés manuellement.

- S'il est en lecture seule, le champ ne pourra pas être modifié dans l'interface.

Ne modifier cette option que pour les champs créés manuellement.

- La case à cocher Indexé est un champ technique utilisé pour l'indexation en base de données. En cochant la case, on crée un index sur la table via ce champ ; ceci accélère les recherches sur les tables volumineuses. L'indexation doit toutefois rester mesurée car la création d'index peut prendre beaucoup d'espace disque.
  Plus le champ est discriminant, plus c'est efficace. Par exemple, un index sur un champ booléen est inutile (car non discriminant).
- L'option copié détermine si la valeur du champ est copiée lors de la duplication d'un enregistrement. Dans le cas contraire, le champ prendra sa valeur par défaut.
- Si le type du champ est float, un champ Précision apparaît dans les Propriétés de base. Il définit la précision décimal du champ.

L'onglet « Propriétés avancées »

Cet onglet contient tous les paramétrages avancés, réservés aux utilisateurs avertis :

- Champ lié : ce champ est réservé à l'éditeur d'Open-Prod ;
- Si l'option Calculer est activée, la valeur du champ est calculée par une formule Python.

Sur l'exemple ci-contre, le champ Date théorique (date_theo) est calculé en ajoutant le délai fournisseur compte tenu du calendrier du fournisseur (et donc des jours ouvrés) à la date du jour.

- Les dépendances sont les champs déclenchant le calcul lors de leur modification. Le calcul est relancé à chaque modification d'une dépendance. Les dépendances sont à séparées par une virgule.

Sur l'exemple ci-contre, une date théorique de disponibilité d'un achat est calculée en fonction du délai du fournisseur. Si je modifie le délai, tous les champs Date théorique existants seront recalculés, même les plus anciens.

- Le champ Stocké détermine si la valeur du champ est stockée en base de données. Le stockage de valeurs est nécessaire pour garder la valeur propagée. Un champ dont la valeur n'est pas stockée est appelé un « champ fonction » : il sera recalculé en permanence.

Seuls les champs stockés sont filtrables dans les listes.

Exemple d'utilisation : on crée un champ « Délai » affiché dans les lignes de devis, qui remonte automatiquement la délai d'engagement renseigné dans l'onglet Vente de la fiche du produit associé. Mais on le stocke pour garder la valeur à la date du devis puisque dans 10 ans, le délai d'engagement ne sera probablement plus le même. On a ainsi l'historique.

- L'option Modifié par onchange doit être cochée pour propager la valeur d'un champ calculé mais pouvoir modifier le résultat.

Exemple : on souhaite propager dans les ventes un champs Pays ajouté manuellement dans le formulaire du client. On souhaite pouvoir le modifier dans certaines ventes.

Définir un champ unique ne se fait pas via ce menu mais par l'ajout d'une contrainte SQL. Il faut donc se connecter à la base de données et y ajouter la contrainte manuellement. Cela peut se faire par cette requête par exemple :

ALTER TABLE tablename ADD CONSTRAINT constraintname UNIQUE (columns);

Quelques exemples

- Champ calculé : l'exemple 1 est un champ calculé en lecture seule dans les ventes. Il est calculé à partir d'un champ de la fiche client et changer la valeur du champ dans le client le modifie également dans la vente. Il n'est pas stocké en base et il sera donc impossible de filtrer sur celui-ci.
- Champ calculé et stocké : l'exemple 2 montre un champ calculé et stocké en base. Les filtres sont alors utilisables.
- Champ calculé par défaut, stocké et modifiable : l'exemple 3 est un champ calculé par défaut avec le champ dans le partenaire mais est tout de même modifiable. De la même manière, changer de partenaire change la valeur du champ Analyse avec celle du champ de la fiche du nouveau partenaire.
- Champ calculé par opérations : l'exemple 4 illustre le calcul de la quantité de produit acheté chez un fournisseur.
- L'exemple 5 permet de calculer le nombre de conditionnements nécessaires à la fabrication d'un lot de produit en fonction du packaging. Ce packaging est remonté de la vente par le même procédé que l'exemple 3 (il est donc modifiable dans l'OF). Le nombre de conditionnement est calculé en se basant sur la quantité à produire et la quantité totale par packaging. C'est pour cela que le champ x_packaging_qty dépend de ces deux champs. Il doit être recalculé au changement de quantité à produire et au changement de packaging.
- L'exemple 6 permet de calculer la somme du montant des ventes terminées pour un client entre 2 dates. On va donc rechercher parmi toutes ventes, celles sont le client correspond à l'id de la fiche partenaire dans laquelle on se trouve. On ne retient que celle terminées et dont la date demandée est entre le 31/12/20 et 01/01/22. Puis on fait la somme.

Une image contenant texteDescription générée automatiquement

Une image contenant texteDescription générée automatiquement

Une image contenant texteDescription générée automatiquement

Exemple 4 : champ de type float calculé

Une image contenant texteDescription générée automatiquement

Une image contenant texteDescription générée automatiquement

Une image contenant texteDescription générée automatiquement

Autre syntaxe possible : Concaténation d'une année et semaine

```
for record in self:

   if record.expected_date:

      record ['x_test_year'] = str(datetime.datetime.strptime(record.expected_date, '%Y-%m-%d').isocalendar()[0])+'-'+str(datetime.datetime.strptime(record.expected_date, '%Y-%m-%d').isocalendar()[1])

   else:

      record ['x_test_year'] = '0'
```

Autres champs possibles : exemple sur le park

Faire un champs fonction dans le park qui donne l'état de l'intervention du park

for record in self:

intervention_rc = self.env['intervention'].search([('park_id', '=', record.id), ('state', 'in', ('waiting', 'affect'))], limit=1)

if intervention_rc:

record['x_test']= intervention_rc.get_selection_field_label('state')

else:

record['x_test']=False

Gestion de champs fonction ou related pour les traductions pour les champs sélection et display state

Champs sélection :

Libellé d'un champ selection: get_selection_field_label

self.partner_id.get_selection_field_label('state')

Champs display state :

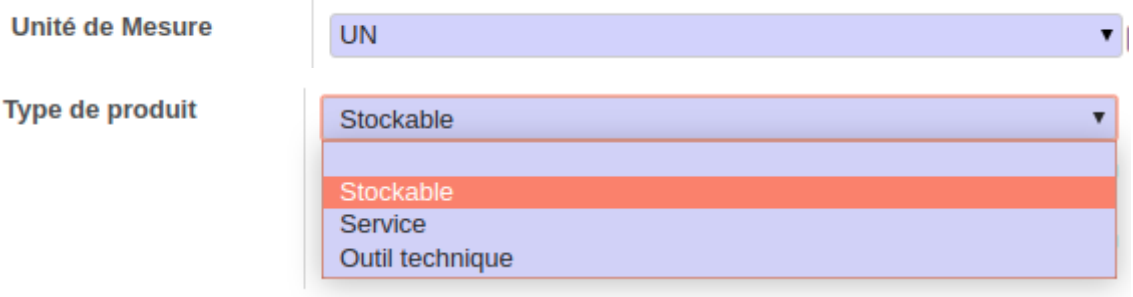
Libellé du diplay_state: get_display_state_label

self.partner_id.get_display_state_label()

Les types de champs

Open-Prod fonctionne avec différents types de champs. Cette page fournit une brève description technique de chaque type, accompagnée d'une illustration.

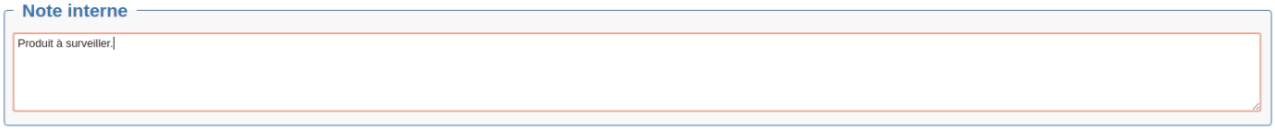
- Many2one : relation N-1 entre deux tables



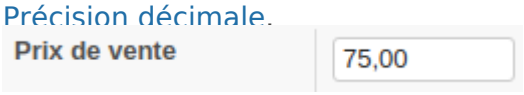
- Booléen : case à cocher. Porte la valeur vrai (True) lorsque la case est cochée, et faux (False) lorsqu'elle est décochée.

- Char : chaîne de caractères (nombre limité)

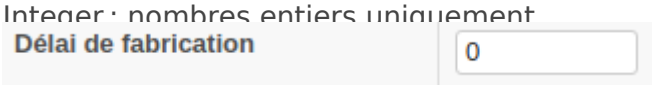
- Texte : chaîne de caractères (nombre illimité).



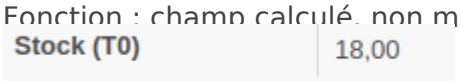
- One2many : relation 1-N entre deux tables.
  Une image contenant texteDescription générée automatiquement
- Float : nombre à virgule. Le nombre de décimales est paramétrable depuis le menu Précision décimale.



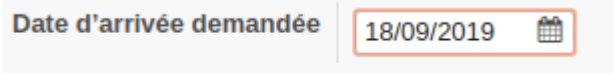
- Integer : nombres entiers uniquement



- Fonction : champ calculé, non modifiable.



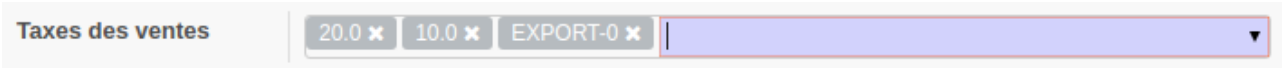
- Date : contient des dates uniquement



- Datetime : date et heure

- Time : champ de durée, le format peut varier d'un champ à l'autre (hh:mm, hh:mm:ss).

- Many2many : relation N-N entre deux tables.



Représentation des modèles

Les représentations de modèle définissent l'affichage d'un objet. On les appelle aussi name_get.

Par défaut, des représentations sont prévues dans le code, mais elles peuvent être modifiées. Nous allons présenter un exemple ci-dessous.

On voit encadrées sur cette image 2 représentations de modèle :

- Les produits sont affichés : [Code Produit] Nom du produit
- Les partenaires sont affichés : [Réf partenaire] Nom du partenaire

Une image contenant texteDescription générée automatiquement

Modification d'une représentation de modèle

Afin de modifier une représentation de modèle, il faut d'abord saisir :

- Le modèle sur lequel elle s'applique.
- La valeur de la nouvelle représentation, écrite en expression python.

Exemple

Ici avec cette valeur : [${object.categ_id.name}] - ${object.code} - ${object.name} , on représente l'objet des produit par [Catégorie de produit] - Code produit - Nom

Une image contenant tableDescription générée automatiquement

Il n'y a pas de limite tant que l'on respecte la syntaxe, en revanche, pour des questions d'ergonomie, d'affichage correct et de performances, il est recommandé de rester raisonnable quant à l'utilisation de ces représentations.

Recherche rapide

La recherche rapide défini les champs sur lesquels on va rechercher un objet. On l'appelle aussi name_search.

Par défaut, les champs de recherche sont imposés dans le code, mais on peut en définir de nouveau en créant une recherche rapide : name_search paramétrable.

Définition d'une recherche rapide

Afin de modifier une recherche rapide, il faut renseigner :

- Le modèle sur lequel elle s'applique.
- Les champs sur lesquels on va rechercher : on a ici accès à tous les champs du modèle saisi précédemment.

Exemple

Définition d'une recherche rapide sur l'objet des produits : Par défaut, on peut rechercher sur le nom ou le code du produit mais on ajoute ici la possibilité de filtrer sur la catégorie du produit.

Ici, le code et le name ont été ajoutés, en plus de la catégorie. Cela n'a pas vraiment d'utilité car ce sont déjà les champs par défaut de recherche. Les champs listés s'ajoutent à ceux déjà défini par défaut dans le code.

Une image contenant tableDescription générée automatiquement

Une image contenant texteDescription générée automatiquement

Onchange manuel

Le onchange est un mécanisme de modification d'un champ lors du changement d'un autre champ dans le formulaire. C'est le mécanisme de propagation que l'on retrouve partout dans le système. Quand on crée une vente, à la sélection du client, l'adresse, les conditions de paiement, ... se remplissent via des onchanges sur le champ client. De la même manière, quand on modifie l' adresse livrée l'ensembles des champs, rue, rue 2, ... sont modifiés en héritant des champs présent dans dans l'adresse. Ce sont tous des onchanges définis par le code source.

Pour augmenter le degré de personnalisation de l'ERP, il est également possible d'en créer manuellement dans l'interface. Cela permet notamment de renseigner automatiquement un champ en agissant sur un autre.

Par exemple, il serait possible de créer un champ « Client final » dans la fiche client et de le remonter automatiquement quand l'on crée un OF et que l'on renseigne le client. Il faudra donc créer un onchange manuel pour modifier la valeur d'un champ existant "Client final" dans les OFs lors du choix du client. Cet exemple est traité dans la page.

La définition des onchanges manuels se fait via le menu Configuration > Gestion avancée de l'administration > Structure de la base de données > Onchange manuel.

Entête

Un onchange manuel est définit ainsi :

- Nom : description du onchange, permet de le différencier des autres.
- Modèle : nom technique de l'objet sur lequel le onchange va s'appliquer.

Afin de trouver ce nom il suffit de passer en mode debug puis de laisser la souris sur un champ, l'objet est alors le modèle recherché.

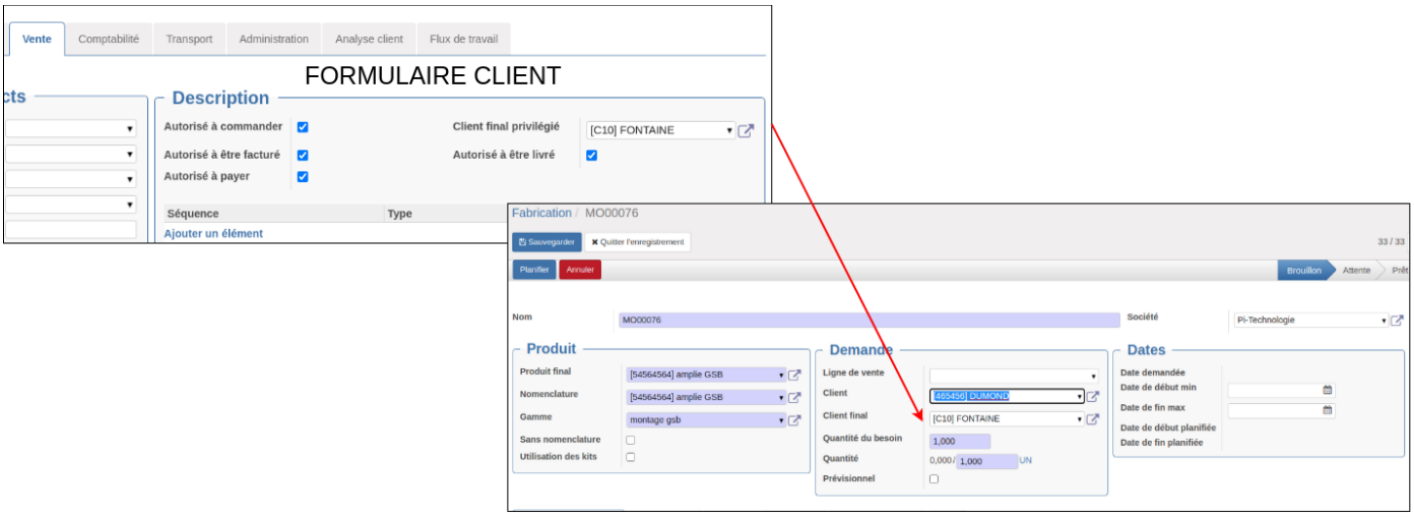
Onglet champs déclencheurs

Dans cette liste il va être possible d'ajouter tous les champs de l'objet sélectionné dans le champ Modèle. Ces champs vont alors déclencher les onchange dès qu'ils seront modifiés.

Une image contenant texteDescription générée automatiquement **Onglet code**

Dans ce cadre il va être possible de rentrer du code python qui va être déclenché dès qu'un des champs déclencheurs est modifié.

Une image contenant texteDescription générée automatiquement



Avertissement via les onchange

Les onchanges manuels permettent également d'ajouter des avertissements sous forme de pop-up dans l'interface de l'utilisateur. Ces avertissements ne seront pas bloquant. En revanche, il est possible de les faire apparaître sur n'importe quel objet : un objet ou même un assistant (affichage temporaire). Pour cela, il faut ajouter un WARNING dans le code python. (Voir ci-dessous).

Aide au remplissage de l'onglet code

Format du résultat :

Valeurs à affecter

Le résultat doit être un dictionnaire nommé "result". Sa clé doit être le nom du champ à modifier. Sa valeur doit être la valeur du champ à modifier

Avertissement

Pour afficher une fenêtre d'avertissement, il faut remplir le message dans une variable nommée "warning"

Variables utilisables :

- object : enregistrement sur lequel le onchange est lancé
- time : bibliothèque Python
- datetime : bibliothèque Python
- re : bibliothèque Python permettant de faire des REGEX

Exemple 1: Traitement sur une chaîne de caractères

```
if object.name:

    result['upper_name'] = object.name.upper()
```

Exemple 2: Calcul de pourcentage

```
if object.total_amount:

result['percentage'] = (object.partial_amount / object.total_amount) * 100

else:

    result['percentage'] = 100.0
```

Exemple 3: Traitement sur une date

```
if object.date:

    result['the_day_after'] = datetime.datetime.strftime(datetime.datetime.strptime(object.date, '%Y-%m-%d') + datetime.timedelta(days=1), '%Y-%m-%d')
```

Exemple 4: Utilisation d'un avertissement

```
if object.percentage < 0:

    warning = 'Impossible de mettre une valeur négative dans le pourcentage'

    result['percentage'] = 0
```

Exemple5: On rapporte dans l'OF les plans de la fiche produit qui sont de type production
Une image contenant texteDescription générée automatiquement

```
res_ids = []

for document_rc in object.product_id.internal_plan_ids:

    if document_rc.type_id.name == 'Production':

        res_ids.append(document_rc.id)

result['internal_plan_ids'] = [(6, 0, res_ids)]
```

Exemple 6 : Remplir le champ description de la ligne de commande d'achat avec le champ description de la fiche produit (traduisible) en fonction de langue du partenaire

Après avoir rendu traduisible le champ description de la fiche produit

```
if object.product_id:

 result['name'] = object.product_id.with_context(lang=object.purchase_order_id.partner_id.lang).description
```

Revision #1
Created 3 November 2023 10:00:58 by Theotim COLIN DE VERDIERE
Updated 3 November 2023 13:29:26 by Theotim COLIN DE VERDIERE