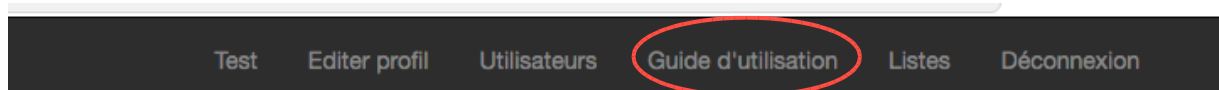


A. Guide du logiciel

Le présent guide repose sur le guide d'utilisation que j'ai écrit pour le logiciel et qui s'y trouve dans le menu inférieur :



Ainsi il se présentera sous la forme de copier-coller du guide et de captures d'écran, avec parfois quelques précisions, en suivant un usager créé sur le logiciel : Mr MICHEL Michel, se trouvant à Aubervilliers.

On rappelle qu'il est possible d'essayer l'application, en se rendant à l'adresse <https://application-pqi-x.herokuapp.com>, où s'y trouve une version de test. On peut s'y connecter avec l'identifiant « Test » et le mot de passe « test123 » pour utiliser le logiciel en mode administrateur. On peut l'utiliser aussi en salarié (« SalTest », « stest123 ») ou en bénévole (« BenevTest », « btest123 »).

Menu inférieur

Utilisateurs

Menu inférieur

On peut y consulter son propre profil, éditer ses informations (nom, prénom, identifiant, mot de passe), voir l'index des utilisateurs, voir le présent guide et se déconnecter.

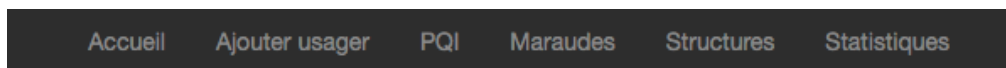
Pour les administrateurs : Listes

On peut y modifier les listes qui apparaissent dans les menus déroulants dans certains formulaires : intervenants, villes, types de rencontre...

C'est le menu de profil utilisateur. Pour les administrateurs, il sert aussi de gestionnaire des comptes utilisateur et des listes de certains menus déroulants.

Menu supérieur

Il s'agit du menu où se trouvent toutes les actions en lien avec les usagers : c'est ici que l'on va pouvoir interagir avec la base de données du logiciel, et mettre à profit ces données.



Ajout d'un usager

On ajoute un usager en remplissant le formulaire d'ajout, en cliquant sur « Ajouter un usager » dans le menu supérieur. On peut ajouter diverses informations que l'on retrouvera sur sa fiche (voir plus bas).

Nouvel usager

On renseignera au moins le nom ou le prénom de l'utilisateur, ainsi que sa ville et son sexe. Sinon, choisir en bas du formulaire le bouton "Inconnu" (il faut néanmoins renseigner au moins sa ville).

Date de naissance : utiliser le bouton "calendrier"

Le formulaire 'Identité' est présenté sur un fond gris. Il contient les champs suivants : 'Nom' avec la valeur 'Michel', 'Prénom' avec la valeur 'Michel', 'Sexe' avec des boutons radio pour 'Mr' (sélectionné) et 'Mme', 'Date de naissance' avec un champ de saisie et un bouton calendrier, et 'Téléphone' avec un champ de saisie.

Après la création de l'utilisateur, on accède à une page d'où l'on peut éditer ses informations complémentaires de suivi (ressources, suivi social, suivi médical, autres informations).

Edition des informations complémentaires de Inconnu(e)

Le formulaire est divisé en quatre onglets : 'Ressources', 'Suivi social', 'Suivi médical' et 'Autres informations'. L'onglet 'Autres informations' est actif. Il contient les sections suivantes : 'Prestations médicales' avec des cases à cocher pour 'CMU', 'CMUC', 'AME' et 'Régime général'; 'Médecin traitant' avec une case à cocher; 'Problématique de santé' et 'Mobilité réduite' avec des cases à cocher; et une section 'Informations complémentaires' en bas.

Ensuite, puisqu'il s'agit d'un ajout, ce logiciel étant conçu pour être utilisé en maraude, il est fort probable qu'une rencontre vient d'avoir lieu avec l'utilisateur ajouté. On est alors redirigé vers une page où l'on peut ajouter une rencontre avec lui (voir plus bas pour les rencontres).

Accueil

Il s'agit de l'index de tous les utilisateurs regroupés par 50 dans un tableau précisant leurs identité, adresse, 5 dernières rencontres, et enfin une case avec des liens rapides pour modifier et supprimer (seulement pour les administrateurs) un utilisateur.

Au passage de la souris, en vert les usagers sur PQI, en rouge ceux qui n'y sont pas.

1 usager

| Identité | Adresse | Dernières rencontres | Edition et Suppression (privilège admin) |
|---|--|---|---|
| Mr MICHEL Michel 01/01/2000 0123456789 Réfèrent : Réfèrent 2 | AUBERVILLIERS 10 rue du Lac Dans une tente | 01/01/16 Ajouter une rencontre | Modifier Supprimer |

Recherche d'un usager et fiche

Sur l'accueil, il est possible de chercher un usager à partir de l'une de ses informations : nom, prénom, sexe, ville, adresse, complément d'adresse, date de naissance, téléphone... Il faut cependant rechercher une information à la fois. En cliquant sur le nom de l'utilisateur recherché, on accède à sa fiche.

On y trouve toutes les informations sur un usager :

- Informations générales (identité, adresse...)
- Appartenance au PQI
- Appartenance à un groupe
- Enfants
- Informations PQI et rencontres : périodes sur PQI, observations PQI, dates de rencontres, signalements et accompagnements
- Statistiques : des chiffres concernant les rencontres avec l'utilisateur
- Informations complémentaires : ressources, suivi social, suivi médical, autres informations
- Fiches de suivi : fiches d'historique des rencontres et fiche de suivi jour, modifiables en cliquant sur leur bouton respectif
- Informations de création : en bas à droite, identifiant de l'utilisateur qui a créé le profil de l'utilisateur ainsi que la date et l'heure de création

Mr MICHEL Michel

Réfèrent : Réfèrent 2
PQI

Informations générales

| Ville | Adresse | Complément d'adresse | Date de naissance | Téléphone |
|---------------|---------------|----------------------|-------------------|------------|
| Aubervilliers | 10 rue du Lac | Dans une tente | 01/01/2000 | 0123456789 |

PQI et interventions

Statistiques

Informations complémentaires

Suivi

Usager ajouté par [Test](#) (28/03/2016, 13:48)

[Supprimer cet usager](#)

Les administrateurs peuvent en plus supprimer l'utilisateur (bouton rouge en bas à droite). Ici, l'utilisateur n'appartient pas à groupe et n'a pas d'enfants. Si c'était le cas, le groupe apparaîtrait en-dessous de « PQI » et un menu cliquable « Enfants » apparaîtrait au-dessus du menu « PQI et Interventions ». D'un simple clic, ces menus se déroulent pour qu'on accède à certaines informations. Par exemple, après un clic sur « PQI et Interventions » :

| Ville | Adresse | Complément d'adresse | Date de naissance | Téléphone |
|---------------|---------------|----------------------|-------------------|------------|
| Aubervilliers | 10 rue du Lac | Dans une tente | 01/01/2000 | 0123456789 |

PQI et interventions

Périodes sur PQI : 28/03/16 - toujours sur le PQI.

Observations PQI : Mr est solitaire.

Dates de rencontre

01/01/16 [Maraude salariés 1] -> Test

Dates de signalement

01/01/16 [Signalement 115]

Dates d'accompagnement

01/01/16 [Accompagnement 115]

[Ajouter une rencontre](#)

[Editer ou supprimer une rencontre](#)

Statistiques

En cliquant sur le nom de l'utilisateur (grand bouton bleu), on accède à une page d'édition de ses informations.

Modification des informations de Mr MICHEL Michel

Référent

Référent 2

☒ PQI

Groupe

Identité

Informations générales

Enfants

PQI

☐ Demande d'hébergement

Enregistrer les modifications

PQI

PQI 1

| | | | | | |
|---------------|------------------|----------|------------------|-------|-------------------|
| Aubervilliers | Aulnay-sous-Bois | Bagnolet | Bobigny | Bondy | Clichy-sous-Bois |
| Coubron | Drancy | Dugny | Epinay-sur-Seine | Gagny | Gournay-sur-Marne |
| | | | | | |

Ville par ville, on peut voir l'ensemble des usagers sur PQI au sein d'un tableau précisant leurs identité, adresse, 5 dernières rencontres, observations PQI. Un bouton à côté du nom de la ville au-dessus du tableau permet d'imprimer rapidement ce dernier.

Aubervilliers 1

| Identité | Adresse | Dernières rencontres | Observations |
|---|---------------------------------|---|-------------------|
| Mr MICHEL Michel 01/01/00 0123456789 Réfèrent : Réfèrent 2 | 10 rue du Lac Dans une tente | 01/01/16 Ajouter une rencontre | Mr est solitaire. |

Rencontre


Le logiciel permet d'ajouter des rencontres avec un usager, en cliquant sur le lien "Ajouter une rencontre" que l'on trouve dans la case "Dernières rencontres" sur l'accueil et dans le PQI, ou sur la fiche usager dans l'onglet "PQI et interventions". On renseigne la date et le type de rencontre obligatoirement, puis des informations supplémentaires le cas échéant : signalement, accompagnement, détails de rencontre, prestation(s) délivrée(s), et options (personne non vue, rencontre prévisionnelle : voir plus bas).

Ajouter une rencontre avec Mr MICHEL Michel

Informations nécessaires

Date

01/04/2016




Type

Quel type de rencontre ? ▾

Ville

Aubervilliers ▾

Signalement

Nouveau signalement 

Il est possible de modifier ou de supprimer une rencontre ajoutée avec un usager à partir de la fiche de celui-ci, dans l'onglet "PQI et interventions", en cliquant sur le lien "Editer ou supprimer une rencontre" puis en renseignant la date et le type de la rencontre (voir la capture d'écran de l'onglet « PQI et Interventions » de la fiche usager).

Editer ou supprimer une rencontre avec Mr MICHEL

Date

01/04/2016



Type

Quel type de rencontre ?

Editer

Supprimer

Modifier la rencontre avec Mr MICHEL Michel

Maraude salariés 1 du 01/01/16

Rencontre créée par Test

Supprimer la rencontre

Informations nécessaires

Date

01/01/2016



Automatiquement, la rencontre s'ajoute à la fiche d'historique des rencontres de l'utilisateur, et le cas échéant aux informations de la maraude sur laquelle l'utilisateur a été vu (s'il s'agit bien d'une maraude).

Suivi

Fiche de rencontres

Fiche de suivi jour

// Maraude salariés 1 [01/01/16] //

Signalement 115

Accompagnement 115

Prestation alimentaire – Duvet

Mr était content de voir le Samu.

Maraude

Sur l'onglet Maraudes, on trouve un index des maraudes par 5 regroupées au sein d'un tableau

précisant le type, la date et les villes parcourues, prévues et parcourues en déroutage (villes modifiables en cliquant sur "Ajouter les villes théoriquement parcourues" ou "Modifier les villes" si elles ont déjà été ajoutées), les rencontres faites, et les signalements et accompagnements sur chaque ville.

Index des maraudes

Nouvelle maraude

Rechercher des maraudes

Rechercher

1 maraude

| Maraude | Rencontres | Signalements et accompagnements |
|--|---------------------------------------|---|
| Maraude salariés 1 01/01/16 / Salarié 1 / Aulnay-sous-Bois <input type="checkbox"/> Bagnolet <input type="checkbox"/> Bobigny <input type="checkbox"/> Déroutage(s) : Aubervilliers | AUBERVILLIERS : - Mr MICHEL Michel | Signalements AUBERVILLIERS : - Mr MICHEL Michel Accompagnements AUBERVILLIERS : - Mr MICHEL Michel |

Il est possible de créer une maraude, mais en ajoutant simplement les rencontres avec chaque usager les maraudes se créent automatiquement.

En cliquant sur la maraude (lien en bleu dans la première case), on accède à la fiche de la maraude. On y trouve toutes les informations sur la maraude :

- ☐ Villes : villes parcourues et prévues, déroutages
- ☐ Interventions : rencontres, personnes non vues sur déplacement, signalements, accompagnements
- ☐ Statistiques : quelques chiffres concernant les interventions et les déplacements
- ☐ Compte-rendu : le compte-rendu de la maraude qui se génère automatiquement à partir des informations rentrées sur les rencontres

Maraude salariés 1 [01/01/16]

/ Salarié 1 /

Villes

Interventions

Statistiques

Compte-rendu

COMPTE-RENDU DE MARAUDE
Maraude salariés 1 [01/01/16]
/ Salarié 1 /

AUBERVILLIERS :
- Mr MICHEL Michel [Signalement 115] [Accompagnement 115] {Prestation alimentaire - Duvet} : Mr était content de voir le Samu.

Supprimer la maraude

Structures

Sur l'onglet Structures, on trouve un index des structures par 20 et où l'on voit en un coup d'oeil en-dessous du nom des structures les nombres de signalements qu'elles ont transmis et d'accompagnements qui y ont été faits.

Index des structures

Nouvelle structure

Rechercher des structures

Rechercher

Le refuge

1 accompagnement

En cliquant sur une structure, on accède à sa page dans laquelle sont détaillés les signalements et les accompagnements (Identité de l'utilisateur, date et type de la maraude).

Le refuge

Rechercher des accompagnements/signalements

Rechercher

Accompagnements 1

| Date | Usager | Maraude |
|----------|------------------|--------------------|
| 01/01/16 | Mr MICHEL Michel | Maraude salariés 1 |

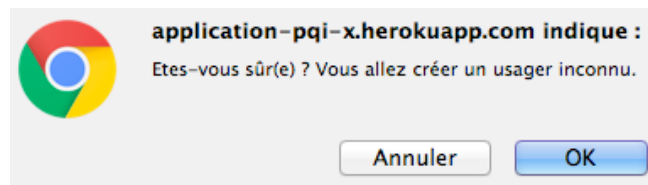
Supprimer la structure

Ajout d'un inconnu

Cette option devient utile lorsque par exemple un usager a été signalé par un tiers qui ne connaît pas son identité, ou lorsqu'un usager rencontré ne veut pas donner d'identité. On ajoute alors la personne en tant qu'inconnue et on peut ajouter une rencontre (prévisionnelle si on en a besoin) avec celle-ci. Si l'on apprend son identité lors d'une maraude, on pourra ensuite accéder à la fiche usager de l'inconnu(e) pour la modifier et renseigner ce qu'on a appris sur l'utilisateur.

Ajouter

Inconnu



Rencontre prévue

Cette option devient utile dans le cas où une personne a été signalée avant la maraude par exemple : on renseigne la rencontre en déclarant qu'elle est prévisionnelle, ce qui signifie qu'elle n'est pas considérée comme réalisée mais comme étant en suspens et à faire.

Sur l'accueil et le PQI, dans la case "Dernières rencontres", on peut voir sous les dernières dates de rencontres le nombre de rencontres prévues avec un usager. En allant sur la fiche de celui-ci, on voit alors en haut l'ensemble des rencontres prévues avec l'usager. On voit aussi les rencontres prévues sur l'accueil du PQI.

Inconnu(e)

PQI

1 Rencontre prévue

- [Maraude salariés 2](#) du 01/01/16 [Signalement tiers]

PQI 2

1 Rencontre prévue

- [Inconnu\(e\) \(Aulnay-sous-Bois\)](#) : [Maraude salariés 2](#) du 01/01/16 [Signalement tiers]

Après avoir réalisé la rencontre, on clique sur celle-ci dans la fiche usager, et il est alors possible de la modifier et de donner les informations collectées lors de la rencontre. Si on a par exemple raté la personne et que l'on veut remettre cette rencontre à plus tard, on peut en changer la date et la déclarer toujours prévisionnelle.

Statistiques

On peut demander au logiciel de calculer certaines statistiques sur une période donnée pour toutes les villes du département ou sur une ville en particulier.

Statistiques

Effectuer une recherche par période ou par ville et période

Début de la période

01/01/2016

Fin

31/01/2016|

Ville

Choisissez une ville

Type

Choisissez un type de rencontre

Valider

Ces statistiques portent sur les rencontres, les déplacements, les prestations.

Statistiques sur le département du 01/01/2016 au 31/01/2016

Entre parenthèses : (Nb d'hommes/de femmes ; d'enfants)

Pour les maraudes : (Salariés/Bénévoles/Jour/Médicale)

Pour les prestations : Alimentaire/Vestiaire/Duvet/Hygiène

Le nombre total de maraudes n'est pas la somme des chiffres de la colonne ! Il s'agit bien du nombre total de maraudes effectuées sur la période

| Ville | Nb maraudes passées sur la ville | Nb total rencontres (PQI + signalés) | Nb personnes différentes rencontrées (PQI + signalés) | Nb non vus sur déplacement (PQI + signalés) | Nb personnes signalées | Nb personnes signalées différentes | Nb signalés non vus | Nb personnes accompagnées | Nb de prestations | Nb demandes d'hébergement formulées |
|------------------|----------------------------------|--------------------------------------|---|---|------------------------|------------------------------------|---------------------|---------------------------|-------------------|-------------------------------------|
| Aubervilliers | 0 | 1 (1/0 ; 0) | 1 (1/0 ; 0) | 0 | 1 (1/0 ; 0) | 1 (1/0 ; 0) | 0 | 1 (1/0 ; 0) | 1/0/1/0 | 0 |
| Aulnay-sous-Bois | 1 (1/0/0/0) | 0 | 0 | 0 | 1 (0/1 ; 0) | 1 (0/1 ; 0) | 0 | 0 | 0 | 0 |
| Bagnolet | 1 (1/0/0/0) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bobigny | 1 (1/0/0/0) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Villetaneuse | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Total | 1 (1/0/0/0) | 1 (1/0 ; 0) | 1 (1/0 ; 0) | 0 (0/0) | 2 (1/1 ; 0) | 2 (1/1 ; 0) | 0 (0/0) | 1 (1/0 ; 0) | 1/0/1/0 | 0 |

Comptes

Il existe trois types de comptes :

- Salarié ;
- Administrateur : ils ont des droits de suppression (usagers, maraudes...) et de modification (listes) supplémentaires ;
- Bénévole : ils ont un accès restreint à des informations sur les usagers. Les informations complémentaires des usagers, les fiches de rencontres et de suivi jour usagers, les compte-rendus des maraudes qui ne sont pas bénévoles leur sont bloqués.

B. Base de données du logiciel

Tables de la base de données

```
extraction-# \dt
```

| List of relations | | | |
|-------------------|-----------------------|-------|--------------|
| Schema | Name | Type | Owner |
| public | enfants | table | clementoriol |
| public | groupes | table | clementoriol |
| public | intervenants | table | clementoriol |
| public | intervenants_maraudes | table | clementoriol |
| public | maraudes | table | clementoriol |
| public | rencontres | table | clementoriol |
| public | schema_migrations | table | clementoriol |
| public | structures | table | clementoriol |
| public | structures_usagers | table | clementoriol |
| public | type_rencs | table | clementoriol |
| public | usagers | table | clementoriol |
| public | users | table | clementoriol |
| public | villes | table | clementoriol |

(13 rows)

La table `schema_migrations` donne juste la version de la base de données en fonction des migrations effectuées.

Détails de chaque table

Notes préliminaires

Chaque objet possède un entier id (toujours clé primaire) et deux attributs `created_at` (date et heure de création) et `updated_at` (date et heure de dernière mise à jour) qui se renseignent automatiquement en Ruby on Rails.

Les relations entre tables se font par référence aux clés id (foreign key) : par exemple un objet « enfant » appartient à un objet « usager » ; dans la table Enfants existe une foreign-key `usager_id` faisant référence à l'id de l'usager auquel « appartient » l'enfant.

Il existe des attributs chaîne de caractère de deux types : string (limité à 255 caractères) et text (sans limitation).

Tous les attributs de type date sont enregistrés dans la base sous la forme `aaaa-mm-jj`. La méthode `strftime` permet de modifier ce format simplement.

Tables

¶ Enfants

extraction=# \d enfants

| Column | Type | Table "public.enfants" | Modifiers |
|----------------|-----------------------------|------------------------|--|
| id | integer | | not null default nextval('enfants_id_seq'::regclass) |
| usager_id | integer | | |
| nom | character varying | | |
| prenom | character varying | | |
| sexe | character varying | | |
| date_naissance | date | | |
| created_at | timestamp without time zone | | not null |
| updated_at | timestamp without time zone | | not null |

Indexes:

"enfants_pkey" PRIMARY KEY, btree (id)

"index_enfants_on_usager_id" btree (usager_id)

Foreign-key constraints:

"fk_rails_e2e26dd4f2" FOREIGN KEY (usager_id) REFERENCES usagers(id)

Les usagers peuvent avoir des enfants. Avec une contrainte foreign-key, chaque objet enfant appartient à un objet usager : il possède un entier `usager_id`, sur lequel il est indexé, faisant référence à quel usager il appartient en donnant l'id de celui-ci. Il possède de plus un nom (string), un prénom (string), un sexe (string : « garçon » ou « fille » ou « »), une date de naissance (date).

¶ Groupes

extraction=# \d groupes

| Column | Type | Table "public.groupes" | Modifiers |
|------------|-----------------------------|------------------------|--|
| id | integer | | not null default nextval('groupes_id_seq'::regclass) |
| nom | character varying | | |
| created_at | timestamp without time zone | | not null |
| updated_at | timestamp without time zone | | not null |

Indexes:

"groupes_pkey" PRIMARY KEY, btree (id)

"index_groupes_on_nom" UNIQUE, btree (nom)

Referenced by:

TABLE "usagers" CONSTRAINT "fk_rails_5ca7fd1119" FOREIGN KEY (groupe_id) REFERENCES groupes(id)

Les usagers peuvent appartenir à un groupe. Chaque groupe possède un nom (string), unique sur index. De plus une contrainte foreign-key lie les usagers qui ont un groupe à leur groupe (un entier `groupe_id` donne l'id du groupe auquel l'usager appartient ; il apparaît dans la table `usagers`).

¶ Intervenants

extraction=# \d intervenants

| Column | Type | Table "public.intervenants" | Modifiers |
|------------|-----------------------------|-----------------------------|---|
| id | integer | | not null default nextval('intervenants_id_seq'::regclass) |
| nom | character varying | | |
| created_at | timestamp without time zone | | not null |
| updated_at | timestamp without time zone | | not null |
| ref | boolean | | |

Indexes:

"intervenants_pkey" PRIMARY KEY, btree (id)

"index_intervenants_on_nom" UNIQUE, btree (nom)

Les intervenants du Samu sont répertoriés : ils apparaissent dans certains menus, ils peuvent être référents d'un usager pour un suivi, ou alors être intervenants en maraude. Chaque intervenant possède un nom (string), unique sur index, et un booléen (`ref`) définissant si un intervenant peut être un référent d'usager ou non (intervenant du pôle jour). Il est ainsi possible de définir des intervenants référents apparaissant spécifiquement dans certains menus déroulants.

□ Intervenants_maraudes

```
extraction-# \d intervenants_maraudes
Table "public.intervenants_maraudes"
  Column      | Type      | Modifiers
-----+-----+-----
maraude_id    | integer   |
intervenant_id | integer   |
Indexes:
  "index_intervenants_maraudes_on_intervenant_id" btree (intervenant_id)
  "index_intervenants_maraudes_on_maraude_id" btree (maraude_id)
```

Les intervenants interviennent sur les maraudes. Une maraude a donc plusieurs intervenants, et chaque intervenant intervient sur plusieurs maraudes. Cette table donne les correspondances, les applications Ruby on Rails utilisent ce système pour faire les liens de dépendance. Une ligne donne l'id d'une maraude avec l'id d'un intervenant : par exemple ci-dessous, l'intervenant avec l'id 1 est intervenu sur la maraude d'id 20, l'intervenant 2 également ; les intervenants d'id 3 et 4 sont intervenus sur la maraude d'id 6, etc.

| maraude_id | intervenant_id |
|------------|----------------|
| 20 | 1 |
| 20 | 2 |
| 6 | 3 |
| 6 | 4 |
| 21 | 3 |

□ Maraudes

```
extraction-# \d maraudes
Table "public.maraudes"
  Column      | Type      | Modifiers
-----+-----+-----
id            | integer   | not null default nextval('maraudes_id_seq'::regclass)
date         | date      |
created_at   | timestamp without time zone | not null
updated_at   | timestamp without time zone | not null
cr           | text      |
type_maraude | character varying |
villes       | text      |
Indexes:
  "maraudes_pkey" PRIMARY KEY, btree (id)
  "index_maraudes_on_date_and_type_maraude" UNIQUE, btree (date, type_maraude)
  "index_maraudes_on_id" UNIQUE, btree (id)
```

Chaque maraude possède une date (date), un type de maraude (string), cette paire d'attributs étant unique sur index (on ne peut trouver deux maraudes ayant les mêmes date et type). Chaque maraude possède de plus un attribut villes (text), donnant la liste des villes censées être parcourues pendant la maraude. Cet attribut contient le nom des villes, chaque ville étant séparé par « \n » définissant un retour à la ligne. L'attribut cr n'est pas utilisé, et l'index unique sur id est inutile mais pas gênant, provenant d'un oubli.

□ Rencontres

| Table "public.rencontres" | | |
|---------------------------|-----------------------------|---|
| Column | Type | Modifiers |
| id | integer | not null default nextval('rencontres_id_seq'::regclass) |
| usager_id | integer | |
| date | date | |
| type_renc | character varying | |
| signale | boolean | |
| signalement | character varying | |
| details | text | |
| created_at | timestamp without time zone | not null |
| updated_at | timestamp without time zone | not null |
| prev | boolean | |
| dnv | boolean | |
| nb_enf | integer | |
| prestas | character varying | |
| accomp | boolean | |
| type_accomp | character varying | |
| ville | character varying | |
| sig_contact | character varying | |
| sig_coords | text | |
| sig_structure | character varying | |
| accomp_structure | character varying | |
| tel | boolean | |
| user_id | integer | |

Indexes:

"rencontres_pkey" PRIMARY KEY, btree (id)
_id, date, type_renc)
"index_rencontres_on_usager_id" btree (usager_id)

Foreign-key constraints:

"fk_rails_75a893ea15" FOREIGN KEY (usager_id) REFERENCES usagers(id)

On définit avec chaque usager des rencontres. Chaque rencontre possède les attributs :

- date (date)
- type_renc (string) : le type de rencontre (maraude salariés 1, maraude bénévoles, rencontre pôle jour...)
- ville (string) : la ville de rencontre
- signale (booléen) : si la rencontre est un signalement ou non
- signalement (string) : le type de signalement le cas échéant (Signalement 115 ou Signalement tiers)
- sig_contact (string) : le nom du contact dans le cas d'un signalement tiers
- sig_coords (text) : les coordonnées du contact dans le cas d'un signalement tiers
- sig_structure (string) : la structure signalante s'il s'agit d'un signalement tiers par une structure
- accomp (booléen) : si la rencontre est un accompagnement
- type_accomp (string) : le type d'accompagnement le cas échéant (Accompagnement 115 ou Accompagnement SIAO ou Autre)
- accomp_structure (string) : la structure où l'on a accompagné l'utilisateur le cas échéant
- details (text) : les commentaires sur la rencontre
- prev (booléen) : si la rencontre est rentrée avant d'avoir été réellement faite, afin de la prévoir en avance, ouvrant à certaines options du logiciel
- dnv (booléen, pour « Déplacement Non Vu ») : si l'on s'est déplacé sans voir la personne
- tel (booléen) : si l'on a eu la personne au téléphone sans se déplacer
- nb_enf (entier) : le nombre d'enfants qui accompagne l'utilisateur rencontré, compris entre 0 et le nombre d'enfants de l'utilisateur renseigné sur le logiciel
- prestas (string) : attribut donnant les prestations délivrées pendant la rencontre, construit sur le même principe que l'attribut villes de la table Maraudes

Chaque rencontre appartient à un usager, l'entier usager_id lui faisant référence par une contrainte foreign-key.

Chaque rencontre est unique sur la base d'un index unique sur le triplet (usager_id, date, type_renc). Deux rencontres différentes ne peuvent avoir été faites avec le même usager, le même jour, sur le même type de rencontre.

- Structures

```
extraction=# \d structures
```

| Column | Type | Table "public.structures" | Modifiers |
|------------|-----------------------------|---------------------------|---|
| id | integer | | not null default nextval('structures_id_seq'::regclass) |
| nom | character varying | | |
| ville | character varying | | |
| adresse | character varying | | |
| created_at | timestamp without time zone | | not null |
| updated_at | timestamp without time zone | | not null |

Indexes:

- "structures_pkey" PRIMARY KEY, btree (id)
- "index_structures_on_nom" UNIQUE, btree (nom)

Chaque structure possède un nom (string) unique sur index, une ville (string), une adresse (string).

□ Structures_usagers

Voir la table Intervenants_maraudes, il s'agit exactement du même système.

□ Type_rencs

```
extraction=# \d type_rencs
```

| Column | Type | Table "public.type_rencs" | Modifiers |
|------------|-----------------------------|---------------------------|---|
| id | integer | | not null default nextval('type_rencs_id_seq'::regclass) |
| nom | character varying | | |
| mar | boolean | | |
| created_at | timestamp without time zone | | not null |
| updated_at | timestamp without time zone | | not null |

Indexes:

- "type_rencs_pkey" PRIMARY KEY, btree (id)
- usager_id | integer |

Indexes:

- "index_structures_usagers_on_structure_id" btree (structure_id)
- "index_structures_usagers_on_usager_id" btree (usager_id)

Chaque type de rencontre possède un nom (string), et un booléen (mar) qui indique si il s'agit d'une rencontre de type maraude ou non. Il est ainsi possible de définir des types de rencontre maraude ou pas, n'apparaissant que dans certains menus déroulants.

□ Usagers

| Column | Type | Modifiers |
|----------------------|-----------------------------|----------------------------------|
| id | integer | not null default nextval('usager |
| s_id_seq'::regclass) | | |
| nom | character varying | |
| prenom | character varying | |
| sexe | character varying | |
| ville | character varying | |
| date_naissance | date | |
| tel | character varying | |
| notes | text | |
| created_at | timestamp without time zone | not null |
| updated_at | timestamp without time zone | not null |
| adresse | character varying | |
| adresse_précis | character varying | |
| user_id | character varying | |
| pqi | boolean | |
| pqi_histo | character varying | |
| fiche | text | |
| groupe_id | integer | |
| ressources | text | |
| montant | double precision | |
| fiche_jour | text | |
| dom | boolean | |
| dom_org | character varying | |
| dom_adr | character varying | |
| tut | boolean | |
| cur | boolean | |
| tutcur_org | character varying | |
| suivi | boolean | |
| suivi_org | character varying | |
| sejour | boolean | |
| cfr | boolean | |
| carte_date | date | |
| autres_infos | text | |
| presta_med | character varying | |
| medecin | character varying | |
| medecin_infos | text | |
| pb_sante | character varying | |
| infos_sante | text | |
| ref | character varying | |
| mobil | boolean | |
| dmde | boolean | |
| date_dmde | date | |
| vu | boolean | |

Indexes:

```
"usagers_pkey" PRIMARY KEY, btree (id)
"index_usagers_on_groupe_id" btree (groupe_id)
"index_usagers_on_nom" btree (nom)
"index_usagers_on_ville" btree (ville)
```

Foreign-key constraints:

```
"fk_rails_5ca7fd1119" FOREIGN KEY (groupe_id) REFERENCES groupes(id)
```

Referenced by:

```
TABLE "rencontres" CONSTRAINT "fk_rails_75a893ea15" FOREIGN KEY (usager_id) REFERENCES usagers(id)
TABLE "enfants" CONSTRAINT "fk_rails_e2e26dd4f2" FOREIGN KEY (usager_id) REFERENCES usagers(id)
```

Chaque usager a pour attributs :

- nom (string)
- prenom (string)
- sexe (string) : « Mr » ou « Mme » ou « »
- ville (string) : les villes du 93 ou « Autre (hors 93) » ou « Ville inconnue »
- date_naissance (date)
- tel (string)
- adresse (string)
- adresse_précis (string) : un complément d'adresse
- user_id : sans contrainte foreign_key pour simplifier, le logiciel ajoute l'id de l'utilisateur qui

- a créé un usager pour le retrouver si besoin
- ❑ pqi (booléen) : si on veut ajouter l'usager au Plan Quotidien d'Intervention (PQI) utilisé par les salariés et bénévoles pendant les maraudes (c'est la liste des usagers à voir sur une maraude)
- ❑ notes (text) : des observations, rapportées sur le PQI
- ❑ pqi_histo (string) : un historique des périodes où l'usager a été sur le PQI
- ❑ fiche (text) : une fiche des rencontres
- ❑ fiche_jour (text) : une fiche de suivi en journée
- ❑ ressources (text) : la liste des ressources d'un usager, fondée sur le même principe que l'attribut villes de la table Maraudes
- ❑ montant (nombre) : le montant des ressources
- ❑ dom (booléen) : si l'usager est domicilié
- ❑ dom_org (string) : l'organisme de domiciliation le cas échéant
- ❑ dom_adr (string) : l'adresse de domiciliation le cas échéant
- ❑ tut (booléen) : si l'usager est sous tutelle
- ❑ cur (booléen) : si l'usager est sous curatelle
- ❑ tutcur_org (string) : l'organisme de tutelle ou de curatelle le cas échéant
- ❑ suivi (booléen) : si l'usager est suivi par un organisme
- ❑ suivi_org (string) : l'organisme de suivi le cas échéant
- ❑ sejour (booléen) : si l'usager possède un titre de séjour
- ❑ cfr (booléen) : si l'usager possède une Carte Nationale d'Identité ou un passeport
- ❑ carte_date (date) : la date d'expiration du titre de séjour, de la CNI ou du passeport le cas échéant
- ❑ autres_infos (text) : informations complémentaires pour le suivi social
- ❑ prestat_med (string) : liste des prestations médicales, fondée sur le même principe que l'attribut villes de la table Maraudes
- ❑ medecin (string) : si l'usager possède un médecin traitant (« Oui » ou « Non » ou « »)
- ❑ medecin_infos (text) : les coordonnées du médecin le cas échéant
- ❑ pb_santé (string) : si il y a problématique de santé (« Oui » ou « Non » ou « »)
- ❑ mobil (boolean) : si l'usager a un problème de mobilité
- ❑ infos_sante (text) : informations complémentaires pour le suivi médical
- ❑ ref (string) : le nom du référent si l'usager en a un (au Samu)
- ❑ dmde (booléen) : si l'usager est en demande d'hébergement
- ❑ date_dmde (date) : la date de la demande le cas échéant
- ❑ vu (booléen) : permet de certifier si un administrateur a bien vu que l'usager a été créé (afin de permettre aux administrateurs de contrôler les nouvelles entrées)

Une contrainte foreign_key sur l'appartenance à un groupe : un entier groupe_id donne l'id du groupe auquel un usager appartient le cas échéant. Les usagers sont indexés séparément sur l'id de leur groupe (s'ils en ont un), sur leur nom et sur leur ville. Les usagers peuvent posséder des enfants et des rencontres (on retrouve une foreign-key contrainte usager_id dans les tables Enfants et Rencontres).

❑ Users

```
extraction-# \d users
```

| Column | Type | Table "public.users" | Modifiers |
|-----------------|-----------------------------|----------------------|--|
| id | integer | | not null default nextval('users_id_seq'::regclass) |
| nom | character varying | | |
| created_at | timestamp without time zone | | not null |
| updated_at | timestamp without time zone | | not null |
| prenom | character varying | | |
| identifiant | character varying | | |
| password_digest | character varying | | |
| admin | boolean | | default false |
| benev | boolean | | |

Indexes:

"users_pkey" PRIMARY KEY, btree (id)

"index_users_on_identifiant" UNIQUE, btree (identifiant)

Cette table ne sera certainement pas utile. Elle contient les utilisateurs du logiciel, qui ont un nom (string), un prenom (string), un identifiant (string) unique sur index, un mot de passe crypté (password_digest, string), un booléen (admin) et un booléen (benev pour bénévole).

▢ Villes

```
extraction-# \d villes
```

| Column | Type | Table "public.villes" | Modifiers |
|------------|-----------------------------|-----------------------|---|
| id | integer | | not null default nextval('villes_id_seq'::regclass) |
| nom | character varying | | |
| created_at | timestamp without time zone | | not null |
| updated_at | timestamp without time zone | | not null |
| ville_93 | boolean | | |

Indexes:

"villes_pkey" PRIMARY KEY, btree (id)

"index_villes_on_nom" UNIQUE, btree (nom)

Chaque ville possède un nom (string) unique sur index, et un booléen (ville_93) définissant si une ville appartient au département de Seine-Saint-Denis ou non. Il est ainsi possible de définir des villes hors 93 n'apparaissant que dans certains menus déroulants.