webrsa	**REUNION WEBRSA - CORRECTIONS DOUBLONS EN BASE DE DONNEES**	ADULLACT Projet
	Webrsa 2.0rc15	
Édition originale	21 Février 2011	***Arnaud AUZOLAT***
Webrsa V2.0rc15		***Chef de projet Webrsa***

Table des matières

Objectifs

Définir les cas de doublons présents en base de données pour chacun des CGs et proposer une solution de suppression de ces doublons.

Liste des cas de doublons et autres anomalies

L'ADULLACT fournit une liste de requêtes détectant un ensemble de cas de doublons.

Ci-dessous les cas détectés jusqu'à présent :

Différentes requêtes	CG66	CG93	CG58
Personnes en doublons	11808	43054	1156
Dossiers sans foyer	0	0	0
Foyers sans aucune personne	15	2	2
Foyers sans demandeur RSA	48	46	4
Foyers sans *adressesfoyers*	332	1	0
Foyers sans *adressesfoyers* de rang 01	332	1	0
Adressesfoyers de rang incorrect	0	0	0
Adressesfoyers en doublons	31	2	0
Adressesfoyers faisant référence au même adresse_id	14079	53408	3870
Adresses sans *adressesfoyers*	0	0	0
Personnes sans prestation RSA	109	10	0
Personnes avec des noms ou des prénoms contenant des caractères inattendus	93	2192	13
Prestations de même nature et de même rôle pour une personne donnée	12	0	0
Prestations de même nature pour une personne donnée	110	0	0
Non demandeurs ou non conjoints RSA possédant des entrées dans les tables métiers (orientation, CER, APREs, …)			
Même personne présente plusieurs fois dans le même bloc dossier RSA du flux XML, avec des informations identiques ou pas	Dépend des flux fournis par la CAF à chaque département		
Même personne mais avec des dates de naissance différentes de quelques jours			
Même personne mais orthographes différentes pour le nom de naissance et le prénom			
Même personne mais champs prénom + 2e prénom différents, ou nom de naissance + prénom (noms composés...)			

Ces résultats ont été obtenus par l'intermédiaire du script shell **anomaliesr.php** présent dans l'application dans *app/vendors/shells/anomaliesr.php.*

Solutions envisagées

Plusieurs solutions sont envisagées face à chacun des cas cités précédemment.

Tout d'abord, nous n'allons traiter, en priorité, que les cas des dossiers dont l'état est « Non clos », i.e. les dossiers dont la valeur **etatdosrsa** vaut :

- 'Z' => 'Non défini',
- '2' => 'Droit ouvert et versable',
- '3' => 'Droit ouvert et suspendu (le montant du droit est calculable, mais l'existence du droit est remis en cause)',
- '4' => 'Droit ouvert mais versement suspendu (le montant du droit n'est pas calculable)'

De plus, nous commencerons par traiter les personnes (Demandeur, Conjoint, ou Autres) vivant dans le même foyer.

Etape 1

Tout d'abord, il est nécessaire d'ajouter en base de données, pour chacune des tables déjà créées ou devant être créées à l'avenir, la notion de ***ON DELETE CASCADE ON UPDATE CASCADE*** sur chacune des FOREIGN KEY.

En effet, ceci nous permettra de supprimer plus facilement une personne lorsque l'on offrira la possibilité de le faire directement à partir de l'application.

Etape 2

Ensuite, il s'agit de supprimer les cas dits « simples » :

1. Nous allons supprimer les personnes qui ne sont ni demandeurs, ni conjoints et qui ont 2 entrées en base de données pour un même foyer

   → liste des *prestations de même nature et de même rôle pour une personne donnée*

2. Nous supprimerons ensuite, les personnes Demandeurs ou Conjoints appartenant à un même foyer mais qui n'ont aucune entrée en base de données notamment pour les tables dites « métiers » (Pas d'orientation, pas de CER, pas de Référent, pas d'APREs …..)

Cas dits plus « complexes »: suppression / fusion

Une fois les solutions précédentes réalisées et afin de ne pas perdre d'informations pour un allocataire donné, nous allons mettre en place au sein de l'application Webrsa (peut-être dans *Administration → Gestion des Doublons*, mais cela reste encore à être défini plus précisément) un lien permettant d'accéder à un écran qui affichera les doublons pour un allocataire donné mais que nous ne pouvons supprimer faute de perdre des informations.

Par exemple :

- Prenons une personne apparaissant 2 fois en tant que demandeur dans un foyer. Une des deux personnes possède une entrée dans la table *orientation* et aucune dans la table *contratsinsertion*. D'un autre côté, l'autre personne demandeur ne possède pas d'entrée dans la table *orientation* mais en possède une dans la table *contratsinsertion* → laquelle doit être supprimée ?
- Autre cas, une personne apparaît deux fois en tant que demandeur dans un foyer. Cette fois-ci, le premier demandeur possède une entrée dans la table *orientation (Orientation vers le social)*. Le deuxième demandeur possède également une entrée dans la table *orientation*, mais l'orientation est différente (*Orientation vers le professionnel*) → laquelle est la bonne ?
- ….

Pour chacune de ces personnes seront affichées les données présentes chez l'une mais non renseignées dans l'autre (et vice versa) et nous offrirons la possibilité de :

- Pouvoir supprimer une des personnes que l'on juge incohérente
- Pouvoir fusionner les données entre les personnes affichées
    - dans ce cas, on fusionne les informations sur une des personnes et on supprime les autres

La main sera laissée aux utilisateurs (ou administrateurs) afin de choisir s'ils souhaitent supprimer une des personnes ou bien s'ils souhaitent fusionner les personnes présentes.

Accès au sein de l'application

Afin de faciliter la recherche d'un doublon en particulier, nous mettrons en place un formulaire de recherche multi-critères pour cibler les doublons de personnes que l'on souhaite corriger, en filtrant :

- par le NIR,
- par le Nom,
- par le Prénom,
- par la Date de naissance,
- … et d'autres informations qui pourront être demandées par les différents départements

Le résultat se présentera sous forme de tableau avec pour chaque personne les différences qui existent (l'affichage des données identiques ne sont pas pertinentes).
Prenons par exemple un allocataire (MR Arnaud AUZOLAT) qui est demandeur dans un foyer avec une entrée dans la table *contratsinsertion* et une entrée dans la table *ressources* avec une DTR du 01/01/2010 au 31/03/2010. Dans ce même foyer, la personne apparaît en double avec comme statut toujours demandeur (Personne B) mais seulement avec une entrée dans la table *ressources* avec une DTR du 01/04/2010 au 30/06/2010. Dans ce cas de figure, nous afficherions un tableau de la forme suivante :

MR Arnaud AUZOLAT							
Personne 1 ◉			Action	Personne 2 ○			Action
Statut	Demandeur		☐Conserver	Statut	Demandeur		☐Conserver
	Date début	Date fin			Date début	Date fin	
Contratsinsertion	01/06/09	01/06/10	☐Conserver	Contratsinsertion			☐Conserver
Ressources	01/01/10	31/03/10	☐Conserver	Ressources	01/04/10	30/06/10	☐Conserver

Pour ce cas de figure, pour l'allocataire en question, nous sommes en présence d'un doublon. A côté de chacun des noms de la *Personne*, un bouton radio sera présent afin de sélectionner lequel des deux (ou + si plusieurs doublons sont présents) sera conservé. Ensuite, pour chacun des champs une case à cocher *Conserver* sera présente fin de décider si on garde les données ou non.
Par défaut, les données non cochées seront supprimées.

Annexe A : Liste des cas de doublons

Exemples de cas de doublons et de proposition de solution.

Dans les exemples qui vont suivre, nous nous sommes basés sur :

- la détection des personnes en doublon pour un même dossier dans un même foyer, dont l'état du dossier se trouve dans un état ouvert, pour une prestation RSA et uniquement pour les demandeurs ou les conjoints.
- les tables possédant une entrée en base. Tout table avec 0 entrée n'est pas prise en compte
- principalement les tables *métiers* car les tables de la CAF pourront toujours être mises à jour suite à la prochaine intégration

  Voici la liste de celles-ci :
  - allocationssoutienfamilial, avispcgpersonnes, conditionsactivitesprealables, contratsinsertion, creancesalimentaires, dsps, grossesses, informationseti, infosagricoles, infospoleemploi, orientsstructs, parcours, personnes_referents, rendezvous, ressources, suivisappuisorientation, titressejour

Les résultats présents dans les tableaux ci-dessous ont été obtenus par l'intermédiaire du script shell **doublons2011.php**, que nous vous fournirons si vous le souhaitez.

Quoiqu'il arrive, il sera présent dans la prochaine version livrée à la mi-Mars.

Tableau décrivant une liste de doublons présents en base CG66 (*seulement un échantillon de 10 cas*)

La tableau a été scindé en deux afin de rentrer dans la page.

Chaque ligne du tableau reprend le nombre d'entrées présent en base par table pour un ID de personne donné (P1.id, P2.id).

Le tableau du haut représente les informations pour le premier ID de la personne en doublon (P1).

Le tableau du bas retrace les informations pour le deuxième ID doublon de la personne (P2).

P1.id	P1.allocationssoutienfamilial	P1.avispcgpersonnes	P1.conditionsactivitespreallables	P1.contratsinsertion	P1.creancesalimentaires	P1.dsps	P1.grossesses	P1.informationseti	P1.infosagricoles	P1.infospoleemploi	P1.orientsstructs	P1.parcours	P1.personnes_referents	P1.rendezvous	P1.ressources	P1.suivisappuiorientation	P1.titressejour	P1.total
54771	0	0	0	2	0	1	0	0	0	4	1	0	2	9	1	0	0	**20**
58898	0	0	0	7	1	1	0	0	0	2	1	0	2	4	1	0	0	**19**
1372	0	0	0	2	1	1	0	0	0	4	1	0	2	4	1	0	0	**16**
579	0	1	0	0	1	1	0	0	0	4	2	0	2	3	1	0	0	**15**
341	0	1	0	2	1	1	0	0	0	0	1	0	1	6	1	0	0	**14**
43699	0	0	0	1	1	1	0	0	0	2	2	0	2	4	1	0	0	**14**
5875	0	0	0	1	1	1	0	0	0	2	1	0	1	5	1	0	0	**13**
54711	0	1	0	1	1	1	0	0	0	3	1	0	2	1	1	0	0	**12**
1338	0	0	0	1	1	1	0	0	0	3	1	0	1	3	1	0	0	**12**
183	0	0	0	1	1	1	0	0	0	3	1	0	1	3	1	0	0	**12**

P2.id	P2.allocationssoutienfamilial	P2.avispcgpersonnes	P2.conditionsactivitespreallables	P2.contratsinsertion	P2.creancesalimentaires	P2.dsps	P2.grossesses	P2.informationseti	P2.infosagricoles	P2.infospoleemploi	P2.orientsstructs	P2.parcours	P2.personnes_referents	P2.rendezvous	P2.ressources	P2.suivisappuiorientation	P2.titressejour	P2.total
108	0	0	0	0	0	1	0	0	0	0	1	0	1	0	0	0	0	**3**
64066	0	0	0	0	1	0	0	0	0	0	1	0	2	0	1	0	0	**5**
40977	0	0	0	0	1	0	0	0	0	0	1	0	0	0	1	0	0	**3**
18823	0	2	0	0	1	0	0	0	0	0	2	0	2	0	1	0	0	**8**
9407	0	2	0	1	1	1	0	0	0	0	1	0	1	0	1	0	0	**8**
1347	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	**2**
73532	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	**2**
408	0	1	0	0	1	1	0	0	0	1	1	0	1	3	1	0	0	**10**
35564	0	0	0	0	1	1	0	0	0	1	1	0	2	0	1	0	0	**7**
20756	0	0	0	0	1	1	0	0	0	0	1	0	1	1	1	0	0	**6**

Prenons l'exemple du dossier « *personnes/view/54711* » pour le CG66

Dans ce dossier, nous sommes en présence d'un doublon sur le demandeur du RSA. La personne apparaît deux fois dans le dossier et le problème vient du fait que cette personne possède plusieurs entrées en dans chacun des 2 cas. En effet, pour la première ligne du tableau, nous pouvons constater que la personne 54771 possède 20 entrées en base et SON doublon, la personne 108, possède 3

entrées en base. Du coup, nous ne pouvons supprimer l'entrée sans vérifier que nous ne perdrons pas d'informations.

- Si on regarde la fiche de la personne au sein de l'application, et que l'on compare les 2 doublons, on peut constater que les données présentes sont identiques, donc le nettoyage de ce doublon ne passera pas par les données de la personne.
- Par contre, selon le tableau ci-dessous, nous constatons que la personne 2 (P2) possède seulement 3 entrées en base, une pour l'Orientation, une pour les DSPs et finalement une pour les référents liés au parcours (*personnes_referents*).
  Du coup, il faut regarder si des conflits n'apparaissent pas si on souhaite réaliser une fusion.
- Hélas, lorsque l'on regarde les DSPs et l'orientation, on peut voir que les informations entre les 2 doublons ne sont pas cohérentes l'une avec l'autre.
  A la même date d'orientation, la personne P1 est orientée vers une structure alors que P2 est orientée vers une structure différente.
  De plus, les DSPs pour les 2 personnes sont également différentes.
- Pour ce cas particulier, nous présenterons les différences entre les 2 personnes dans un tableau comparatif et pour chacune des lignes différentes, nous offrirons le choix de cocher une case précisant si on conserve ou non la valeur.

Tableau décrivant une liste de doublons présents en base CG93 (*seulement un échantillon de 10 cas*)

La tableau a été scindé en deux afin de rentrer dans la page.

Chaque ligne du tableau reprend le nombre d'entrées présent en base par table pour un ID de personne donné (P1.id, P2.id).

Le tableau du haut représente les informations pour le premier ID de la personne en doublon (P1).

Le tableau du bas retrace les informations pour le deuxième ID doublon de la personne (P2).

P1.id	P1.allocationssoutienfamilial	P1.apres	P1.avispcgpersonnes	P1.conditionsactivitespreallables	P1.contratsinsertion	P1.creancesalimentaires	P1.dsps	P1.grossesses	P1.informationseti	P1.infosagricoles	P1.orientations	P1.orientsstructs	P1.parcours	P1.propospdos	P1.ressources	P1.suivisappuiorientation	P1.titressejour	P1.tmporientstructs	P1.total
396897	0	1	1	0	3	1	1	0	0	0	0	1	0	0	1	0	0	0	**9**
210410	0	1	0	0	3	1	1	1	0	0	0	1	0	0	1	0	0	0	**9**
497390	0	0	0	0	0	5	1	1	0	0	0	1	0	0	1	0	0	0	**9**
490971	0	0	0	0	5	1	0	0	0	0	0	1	0	0	1	0	0	1	**9**
483061	0	0	0	0	0	9	0	0	0	0	0	0	0	0	0	0	0	0	**9**
480955	1	0	0	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	**9**
499117	0	0	0	0	0	9	0	0	0	0	0	0	0	0	0	0	0	0	**9**
499117	0	0	0	0	0	9	0	0	0	0	0	0	0	0	0	0	0	0	**9**
327518	0	1	0	0	2	1	0	1	0	0	0	1	0	0	1	0	0	1	**8**
516226	0	1	0	0	1	1	1	0	0	0	0	1	1	0	1	1	0	0	**8**

P2.id	P2.allocationssoutienfamilial	P2.apres	P2.avispcgpersonnes	P2.conditionsactivitespreallables	P2.contratsinsertion	P2.creancesalimentaires	P2.dsps	P2.grossesses	P2.informationseti	P2.infosagricoles	P2.orientations	P2.orientsstructs	P2.parcours	P2.propospdos	P2.ressources	P2.suivisappuiorientation	P2.titressejour	P2.tmporientstructs	P2.total
641480	0	0	1	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	**4**
632607	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	**2**
544052	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	**2**
572311	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	**2**
523580	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	**1**
497642	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**
536924	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
488246	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
507496	0	0	1	0	1	1	0	1	0	0	0	0	0	0	1	0	0	0	**5**
554016	0	1	1	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	**4**

Prenons l'exemple du dossier « *personnes/view/499117»* pour le CG93

Dans ce cas, nous sommes en présence d'une personne présente 3 fois en base de données.

En effet, l'allocataire est présent en tant qu'enfant dans le même dossier mais avec 3 IDs différents (*499117, 536924 et 488246*).

Néanmoins, pour ce cas, nous pouvons constater que les entrées en base sont seulement présentes pour la personne P1 (499117).

Si on regarde dans l'application les informations de la personne en question, on constate que les 3 personnes ont des différences notamment au niveau:

- du Nom de naissance → dans les 3 cas il est différent (une lettre modifiée, un nom plus long)
- de la certification de l'état civil de la personne
- de la commune de naissance

Du coup, nous proposerons de fusionner les informations de ces personnes en une seule. Cette fusion devra être réalisée manuellement car il est important de savoir quelles données sont pertinentes et d'autres non.

Prenons l'exemple du dossier « *personnes/view/327518»* pour le CG93

Cet allocataire apparaît deux fois au sein du même dossier (*327518 et 507496*).

A première vue, dans l'application, on peut constater des différences au niveau de la personne :

- Nom de naissance → une lettre diffère
- Commune de naissance
- Certification de l'état civil de la personne

La différence la plus pertinente par la suite est le fait que ces deux personnes possèdent une entrée dans la table *contratsinsertion*.

La personne P1 possède deux CER et la personne P2 en possède un CER.

Après vérification dans l'application, il semble que les dates de début et de fin des CER ne se chavauchent pas donc il sera facile de fusionner les données en déplaçant le CER le plus vieux (celui de P2 01/11/2009 – 31/01/2010) vers P1. A notre charge de faire en sorte que le CER signé en premier (celui de P2) soit de rang *Premier contrat* et que les deux suivants passent en *Renouvellement.*

Tableau décrivant une liste de doublons présents en base CG58 (*seulement un échantillon de 10 cas*)

La tableau a été scindé en deux afin de rentrer dans la page.

Chaque ligne du tableau reprend le nombre d'entrées présent en base par table pour un ID de personne donné (P1.id, P2.id).

Le tableau du haut représente les informations pour le premier ID de la personne en doublon (P1).

Le tableau du bas retrace les informations pour le deuxième ID doublon de la personne (P2).

P1.id	P1.actionscandidats_personnes	P1.allocationsoutienfamilial	P1.avispcgpersonnes	P1.contratsinsertion	P1.creancesalimentaires	P1.dsps	P1.grossesses	P1.orientsstructs	P1.personnes_referents	P1.propospdos	P1.rendezvous	P1.ressources	P1.titressejour	P1.total
44317	0	1	0	0	1	0	0	0	0	0	0	1	0	3
39464	0	0	1	0	0	0	1	0	0	0	0	1	0	3
44235	0	1	0	0	1	0	0	0	0	0	0	1	0	3
45202	0	1	0	0	1	0	0	0	0	0	0	1	0	3
3544	0	1	0	0	1	0	0	0	0	0	0	1	0	3
3543	0	1	0	0	1	0	0	0	0	0	0	1	0	3
47004	0	1	0	0	1	0	0	0	0	0	0	1	0	3
39418	0	1	0	0	1	0	0	0	0	0	0	1	0	3
47810	0	1	0	0	1	0	0	0	0	0	0	1	0	3
44323	0	1	0	0	1	0	0	0	0	0	0	1	0	3

P2.id	P2.actionscandidats_personnes	P2.allocationsoutienfamilial	P2.avispcgpersonnes	P2.contratsinsertion	P2.creancesalimentaires	P2.dsps	P2.grossesses	P2.orientsstructs	P2.personnes_referents	P2.propospdos	P2.rendezvous	P2.ressources	P2.titressejour	P2.total
47004	0	1	0	0	1	0	0	0	0	0	0	1	0	3
492	0	0	1	0	1	0	0	0	0	0	0	1	0	3
39418	0	1	0	0	1	0	0	0	0	0	0	1	0	3
44323	0	1	0	0	1	0	0	0	0	0	0	1	0	3
47811	0	1	0	0	1	0	0	0	0	0	0	1	0	3
47810	0	1	0	0	1	0	0	0	0	0	0	1	0	3
44317	0	1	0	0	1	0	0	0	0	0	0	1	0	3
44235	0	1	0	0	1	0	0	0	0	0	0	1	0	3
3543	0	1	0	0	1	0	0	0	0	0	0	1	0	3
45202	0	1	0	0	1	0	0	0	0	0	0	1	0	3

Prenons l'exemple du dossier « *personnes/view/39464»* pour le CG58

Dans ce cas, nous sommes en présence d'un demandeur présent deux fois dans un même dossier (*39464 et 492*).

Si on regarde dans l'application, au niveau des informations de la personne, on remarque des différences :

- au niveau du Nom → Le nom est composé et une inversion des positions du nom est présente

- au niveau du 2ème prénom → la valeur est présente dans un seul des 2 cas (P1)
- au niveau de la Commune de naissance → la valeur est présente dans un seul des 2 cas (P1)
- au niveau du certificat de l'état civil de la personne
- au niveau de la grossesse → une entrée est présente pour la personne P1

A première vue, il semblerait qu'une fusion des deux personnes soit réalisable vers la personne P1.

Remarque :

Ces tableaux nous permettent de détecter les différents cas de doublons que nous pouvons remonter.

Pour ces cas, le but est de réaliser une fusion des informations (quand cela est possible), par exemple lorsque des informations telles que les ressources sont entrées dans un des doublons et en parallèle dans l'autre doublon, mais que les dates ne se chevauchent pas, une simple fusion des deux personnes sera réalisable.

Néanmoins, si nous nous trouvons face à des cas où un CER est entré dans un des doublons et un autre est saisi, en parallèle sur le doublon, avec des dates se chevauchant, il faudra réaliser une suppression / fusion manuelle afin d'être sûr de ne pas se tromper.

Annexe B : Requêtes SQL de détection de doublons

Voici la liste des requêtes utilisées pour la détection des doublons.

Les résultats des requêtes présentés ci-dessous ont été obtenus sur les bases de données de chacun des CGs, après mise à niveau des bases jusqu'à la version 2.0rc15 :

- CG66 → dump de la base de reçu par ADULLACT le 15/02/2011
- CG93 → dump de la base de production reçu par ADULLACT le 15/02/2011
- CG58 → dump de la base test reçu par ADULLACT le 24/02/2011

Requête 1 : *Nombre de personnes en doublon détectées*

```
SELECT DISTINCT(p1.id)
        FROM  personnes p1,
                personnes p2
        WHERE p1.id < p2.id
        AND
        (
                ( LENGTH(TRIM(p1.nir)) = 15 AND p1.nir = p2.nir )
                OR ( p1.nom = p2.nom AND p1.prenom = p2.prenom AND p1.dtnai = p2.dtnai )
        )
```

Résultat par CGs :

CG66 → 11808 CG93 → 43054 CG58 → 1156

Requête 2 : Liste *des personnes en doublon dans le même foyer, avec nomnai+prénom+dtnai+rgnai identiques*

```
SELECT
        Personne.id,
        Personne.nomnai,
        Personne.prenom,
        Personne.dtnai,
        Personne.rgnai,
        Personne.foyer_id
FROM
        personnes as Personne,
        personnes as Personne2
WHERE
        Personne.nomnai LIKE Personne2.nomnai
        AND Personne.prenom LIKE Personne2.prenom
        AND Personne.dtnai = Personne2.dtnai
        AND Personne.rgnai = Personne2.rgnai
        AND Personne.foyer_id = Personne2.foyer_id
        AND Personne2.id < Personne.id
;
```

Résultat par CGs :

CG66 → 82 (~0,7% des doublons détectés)	CG93 → 13 (~0,03% des doublons détectés)	CG58 → 15 (~1,3% des doublons détectés)

Requête 3 : Liste *des personnes en doublons dans le même foyer, avec nomnai+prenom+dtnai identiques*

```
SELECT
	Personne.id,
	Personne.nomnai,
	Personne.prenom,
	Personne.dtnai,
	Personne.foyer_id
FROM
	personnes as Personne,
	personnes as Personne2
WHERE
	Personne.nomnai LIKE Personne2.nomnai
	AND Personne.prenom LIKE Personne2.prenom
	AND Personne.dtnai = Personne2.dtnai
	AND Personne.foyer_id = Personne2.foyer_id
	AND Personne2.id < Personne.id
;
```

Résultat par CGs :

CG66 → 1475 (~12,5% des doublons détectés)	CG93 → 34 (~0,08% des doublons détectés)	CG58 → 15 (~1,3% des doublons détectés)

Requête 4 : *Nombre de personnes en doublons dans le même foyer, avec (nom+prenom+dtnai+rgnai) identiques ou bien (nomnai+prenom+dtnai+rgnai) identiques*

```
SELECT
	count(Personne.id)
FROM
	personnes as Personne,
	personnes as Personne2,
	foyers as Foyer
WHERE
	(
		Personne.nomnai LIKE Personne2.nomnai
	OR
		Personne.nom LIKE Personne2.nom
	)
	AND Personne.prenom LIKE Personne2.prenom
	AND Personne.dtnai = Personne2.dtnai
	AND Personne2.id < Personne.id
	AND Personne.foyer_id = Foyer.id
	AND Personne2.foyer_id = Foyer.id
;
```

Résultat par CGs :

CG66 → 3461 (~29% des doublons détectés)	CG93 → 5854 (~13,6% des doublons détectés)	CG58 → 53 (~4,6% des doublons détectés)

Requête 5 : *Nombre de demandeur / conjoint du RSA, en doublons dans le même foyer, avec (nom+prenom+dtnai+rgnai) identiques ou bien (nomnai+prenom+dtnai+rgnai) identiques.*

```
SELECT
	count(Personne.id)
FROM
	personnes as Personne,
	personnes as Personne2,
	foyers as Foyer,
	prestations as Prestation
WHERE
	(
		Personne.nomnai LIKE Personne2.nomnai
	OR
		Personne.nom LIKE Personne2.nom
	)
	AND Personne.prenom LIKE Personne2.prenom
	AND Personne.dtnai = Personne2.dtnai
	AND Personne.id < Personne2.id
	AND Personne.foyer_id = Foyer.id
	AND Personne2.foyer_id = Foyer.id
	AND
	(
		Prestation.personne_id = Personne.id
	OR
		Prestation.personne_id = Personne2.id
	)
	AND Prestation.natprest LIKE 'RSA'
	AND Prestation.rolepers IN ( 'DEM', 'CJT' )
;
```

Résultat par CGs :

CG66 → 5528 (~46,8% des doublons détectés)	CG93 → 6467 (~15% des doublons détectés)	CG58 → 64 (~5,53% des doublons détectés)

Requête 6 : Liste *des NON demandeurs / conjoints du RSA, en doublons dans le même foyer, avec (nom ou prenom ou nomnai)+dtnai identiques*

```
SELECT
	Personne.id,
	Personne2.id,
	Personne.nomnai,
	Personne2.nomnai,
	Personne.nom,
	Personne2.nom,
	Personne.prenom,
	Personne2.prenom,
	Personne.dtnai,
	Personne2.dtnai,
```

```
        Personne.foyer_id,
        Prestation.natprest,
        Prestation2.natprest,
        Prestation.rolepers,
        Prestation2.rolepers
FROM
        personnes as Personne INNER JOIN prestations as Prestation ON ( Personne.id =
Prestation.personne_id ),
        personnes as Personne2 INNER JOIN prestations as Prestation2 ON ( Personne2.id =
Prestation2.personne_id )
WHERE
        (
                Personne.nom LIKE Personne2.nom
        OR
                Personne.nomnai LIKE Personne2.nomnai
        OR
                Personne.prenom LIKE Personne2.prenom
        )
        AND Personne.dtnai = Personne2.dtnai
        AND Personne.id < Personne2.id
        AND Prestation.personne_id < Prestation2.personne_id
        AND Personne.foyer_id = Personne2.foyer_id
        AND Prestation.natprest = 'RSA'
        AND Prestation.rolepers NOT IN ( 'DEM', 'CJT' )
        AND Prestation2.natprest = 'RSA'
;
```

Résultat par CGs :

CG66 → 2015 (~17% des doublons détectés)	CG93 → 15524 (~36% des doublons détectés)	CG58 → 318 (~27,5% des doublons détectés)

Requête 7 : *Recherche des doublons dans les dossiers non clos, pour les personnes ayant un droit ouvert et versable, pour les prestations RSA et dans un même foyer.*

```
SELECT
        COUNT(*)
FROM
        personnes as Personne
                INNER JOIN prestations as Prestation ON ( Personne.id = Prestation.personne_id )
                INNER JOIN calculsdroitsrsa as Calculdroitrsa ON ( Personne.id =
Calculdroitrsa.personne_id )
                INNER JOIN foyers as Foyer ON ( Personne.foyer_id = Foyer.id )
                INNER JOIN dossiers as Dossier ON ( Dossier.id = Foyer.dossier_id )
                INNER JOIN situationsdossiersrsa as Situationdossierrsa ON ( Dossier.id =
Situationdossierrsa.dossier_id ),
        personnes as Personne2 INNER JOIN prestations as Prestation2 ON ( Personne2.id =
Prestation2.personne_id )
WHERE
        (
                Personne.nom LIKE Personne2.nom
        OR
                Personne.nomnai LIKE Personne2.nomnai
        OR
                Personne.prenom LIKE Personne2.prenom
        )
```

```
        AND Personne.dtnai = Personne2.dtnai
        AND
        (
                (
                        Personne.rgnai IS NULL
                AND
                        Personne2.rgnai IS NULL
                )
        OR
                Personne.rgnai = Personne2.rgnai
        )
        AND Personne.id < Personne2.id
        AND Personne.foyer_id = Personne2.foyer_id
        AND Prestation.natprest = 'RSA'
        AND Prestation2.natprest = 'RSA'
        AND Calculdroitrsa.toppersdrodevorsa = '1'
        AND Situationdossierrsa.etatdosrsa IN ( 'Z', '2', '3', '4' )
;
```

Résultat par CGs :

CG66 → 2299 (~19,47% des doublons détectés)	CG93 → 4722 (~11% des doublons détectés)	CG58 → 98 (~8,47% des doublons détectés)

Requête 8 : *Recherche des personnes en doublon dans les dossiers non clos, avec droit ouvert et versable, les 2 rangs de naissance sont nuls ou identiques, les prestations sont RSA , (nom ou prenom ou nomnai)+ dtnai sont identiques*

```
SELECT
        COUNT(*)
FROM
        personnes as Personne
                INNER JOIN prestations as Prestation ON ( Personne.id = Prestation.personne_id )
                INNER JOIN calculsdroitsrsa as Calculdroitrsa ON ( Personne.id =
Calculdroitrsa.personne_id )
                INNER JOIN foyers as Foyer ON ( Personne.foyer_id = Foyer.id )
                INNER JOIN dossiers as Dossier ON ( Dossier.id = Foyer.dossier_id )
                INNER JOIN situationsdossiersrsa as Situationdossierrsa ON ( Dossier.id =
Situationdossierrsa.dossier_id ),
        personnes as Personne2
                INNER JOIN prestations as Prestation2 ON ( Personne2.id =
Prestation2.personne_id )
                INNER JOIN calculsdroitsrsa as Calculdroitrsa2 ON ( Personne2.id =
Calculdroitrsa2.personne_id )
                INNER JOIN foyers as Foyer2 ON ( Personne2.foyer_id = Foyer2.id )
                INNER JOIN dossiers as Dossier2 ON ( Dossier2.id = Foyer2.dossier_id )
                INNER JOIN situationsdossiersrsa as Situationdossierrsa2 ON ( Dossier2.id =
Situationdossierrsa2.dossier_id )
WHERE
        (
                Personne.nom LIKE Personne2.nom
        OR
                Personne.nomnai LIKE Personne2.nomnai
        OR
                Personne.prenom LIKE Personne2.prenom
```

```
        )
        AND Personne.dtnai = Personne2.dtnai
        AND
        (
                (
                        Personne.rgnai IS NULL
                AND
                        Personne2.rgnai IS NULL
                )
        OR
                Personne.rgnai = Personne2.rgnai
        )
        AND Personne.id < Personne2.id
        AND Personne.foyer_id = Personne2.foyer_id
        AND Prestation.natprest = 'RSA'
        AND Prestation2.natprest = 'RSA'
        AND Calculdroitrsa.toppersdrodevorsa = '1'
        AND Situationdossierrsa.etatdosrsa IN ( 'Z', '2', '3', '4' )
        AND Calculdroitrsa2.toppersdrodevorsa = '1'
        AND Situationdossierrsa2.etatdosrsa IN ( 'Z', '2', '3', '4' )
;
```

Résultat par CGs :

CG66 → 1731 (~14,7% des doublons détectés)	CG93 → 3070 (~7,3% des doublons détectés)	CG58 → 65 (~5,62% des doublons détectés)

Requête 9 : *Nombre de foyers en doublons saisis par l'intermédiaire de la préconisation d'orientation*

```
SELECT
   count( DISTINCT(f1.id))
FROM
   personnes as p1
      INNER JOIN foyers as f1 ON ( p1.foyer_id = f1.id )
      INNER JOIN situationsdossiersrsa as s1 ON ( f1.dossier_id = s1.dossier_id ),
   personnes as p2
      INNER JOIN foyers as f2 ON ( p2.foyer_id = f2.id )
      INNER JOIN situationsdossiersrsa as s2 ON ( f2.dossier_id = s2.dossier_id )
WHERE
   (
      ( nir_correct( p1.nir ) AND p1.nir = p2.nir AND p1.dtnai = p2.dtnai )
      OR
      (
         TRIM( BOTH ' ' FROM p1.nom ) = TRIM( BOTH ' ' FROM p2.nom )
         AND TRIM( BOTH ' ' FROM p1.prenom ) = TRIM( BOTH ' ' FROM p2.prenom )
         AND p1.dtnai = p2.dtnai
      )
   )
   AND p1.id < p2.id
   AND f1.dossier_id <> f2.dossier_id
   AND s1.etatdosrsa = 'Z'
   AND s2.etatdosrsa <> 'Z'
;
```

Résultat par CGs :

CG66 → 158 (~1,3% des doublons détectés)	CG93 → 0	CG58 → 0

Requête 10 : *Nombre de foyers saisis par l'intermédiaire de la préconisation d'orientation et ne possédant pas de doublons*

```
SELECT
   count( DISTINCT(Foyer.id))
--     DISTINCT(Foyer.id)
FROM
   personnes as Personne
      INNER JOIN foyers as Foyer ON ( Personne.foyer_id = Foyer.id )
      INNER JOIN situationsdossiersrsa as Situationdossierrsa ON ( Foyer.dossier_id =
Situationdossierrsa.dossier_id )
WHERE
   Situationdossierrsa.etatdosrsa = 'Z'
   AND
   Personne.id NOT IN (
      SELECT
         p1.id
         FROM
            personnes as p1
               INNER JOIN foyers as f1 ON ( p1.foyer_id = f1.id )
               INNER JOIN situationsdossiersrsa as s1 ON ( f1.dossier_id = s1.dossier_id ),
            personnes as p2
               INNER JOIN foyers as f2 ON ( p2.foyer_id = f2.id )
               INNER JOIN situationsdossiersrsa as s2 ON ( f2.dossier_id = s2.dossier_id )
         WHERE
            (
               ( nir_correct( p1.nir ) AND p1.nir = p2.nir AND p1.dtnai = p2.dtnai )
               OR
               (
                  TRIM( BOTH ' ' FROM p1.nom ) = TRIM( BOTH ' ' FROM p2.nom )
                  AND TRIM( BOTH ' ' FROM p1.prenom ) = TRIM( BOTH ' ' FROM p2.prenom )
                  AND p1.dtnai = p2.dtnai
               )
            )
            AND p1.id <> p2.id
            AND f1.dossier_id <> f2.dossier_id
            AND s1.etatdosrsa = 'Z'
            AND s2.etatdosrsa <> 'Z'

   );
```

Résultat par CGs :

CG66 → 278 (~2,3% des doublons détectés)	CG93 → 0	CG58 → 0

Autre script de détection

Un nouveau script a été développé ce jour concernant la détection des doublons.
dev_doublons.php

Un aperçu des résultats de ce script est présenté ci-dessous.

```
Shell de vérification de doublons, cg66_rc15_20110225

---------------------------------------------------------------
<strong>app/vendors/shells/dev_doublons.php</strong> (line <strong>98</strong>)
<pre class="cake-debug">
Array
(
   [Nb de personnes] => 98826
   [Nombre de DEM ou CONJOINT RSA] => 56687
   [Nombre de NON demandeur ou conjoint RSA] => 42071
   [Nombre de personnes sans prestations RSA] => 109
   [Nombre de personnes avec plusieurs prestations RSA] => 54
)

</pre>
<strong>app/vendors/shells/dev_doublons.php</strong> (line <strong>240</strong>)
<pre class="cake-debug">
Array
(
   [Nb de DEM / CJT en doublons dans le même foyer avec même (NIR + DTNAI)] => 0

   [Nb de DEM/CJT en doublons dans le même foyer avec même (Nom+Prénom+DTNAI)] => 9

   [Nb de DEM/CJT en doublons dans le même foyer avec
        Nom différent, même Prénom, même Nom de naissance
        OU
        même Nom, Prénom différent, même Nom de naissance
        OU
        même Nom, même Prénom, Nom de naissance différente
        ET même ( Date de naissance + Rang de naissance )
    ]    => 45

   [Nb de NON DEM / CJT en doublons dans le même foyer avec même (NIR + DTNAI)] => 0

   [Nb de NON DEM/CJT en doublons dans le même foyer avec même (Nom+Prénom+DTNAI)]]
        => 7

   [Nb de NON DEM/CJT en doublons dans le même foyer avec
        Nom différent, même Prénom, même Nom de naissance
        OU
        même Nom, Prénom différent, même Nom de naissance
        OU
        même Nom, même Prénom, Nom de naissance différente
        ET même ( Date de naissance + Rang de naissance )
    ]    => 20

   [Nb de DEM / CJT en doublons dans des foyers différents avec droit ouvert avec même (NIR
+ DTNAI)] => 0

   [Nb de DEM / CJT en doublons dans des foyers différents avec droit ouvert avec même (Nom
+ Prénom + DTNAI)] => 29
```

```
   [Nb de DEM/CJT en doublons dans des foyers différents avec droit ouvert et
         Nom différent, même Prénom, même Nom de naissance
         OU
         même Nom, Prénom différent, même Nom de naissance
         OU
         même Nom, même Prénom, Nom de naissance différente
         ET même ( Date de naissance + Rang de naissance )
    ]    => 67

   [Nb de NON DEM / CJT en doublons dans des foyers différents avec droit ouvert et même
(NIR + DTNAI)] => 0

   [Nb de NON DEM / CJT en doublons dans des foyers différents avec droit ouvert et même
(Nom + Prénom + DTNAI)] => 9

   [Nb de NON DEM/CJT en doublons dans des foyers différents avec droit ouvert et
         Nom différent, même Prénom, même Nom de naissance
         OU
         même Nom, Prénom différent, même Nom de naissance
         OU
         même Nom, même Prénom, Nom de naissance différente
         ET même ( Date de naissance + Rang de naissance )
    ]     => 18

   [Nombre total de DEM / CJT selon les critères précédents] => 150
   [Nombre total de NON DEM / CJT selon les critères précédents] => 54
   [Nombre total de personnes correspondant aux critères précédents] => 204
)

</pre>
```

Annexe C : Proposition d'algorithme d'intégration flux CNAF

Une fois les doublons épurés au maximum, par la suite, il faut éviter de réintégrer ces doublons en base de données.

Nous allons mettre en place des contraintes en base de données afin d'éviter :

- d'avoir des NIR mals formatés,
- d'avoir plus d'1 demandeur ou conjoint par dossier,
- …

Pour cela, nous allons proposer un algorithme au CG93 afin qu'ils puissent faire évoluer les jobs d'intégration, ou plus précisément, afin de solidifier les données stockées en base.

Proposition de nouvel algorithme d'insertion des jobs, afin d'éviter une nouvelle création de doublons lors des prochaines intégrations (A débattre)

```
SI le NUMDEMRSA présent dans le flux existe en base de données

        ALORS SI ( la balise DEM est présente dans le flux

            SI ( le DEM existe en base pour le NUMDEMRSA du flux

                ALORS ( on vérifie le bon état du NIR )
                    SI NIR = OK (
                        ALORS on vérifie SI (
                            ( NIR + DTNAI ) du flux == ( NIR + DTNAI ) en BDD pour le DEM du NUMDEMRSA
                            SI ( OK )
                            ALORS ( UPDATE )
                        )
                    )
                    SINON on vérifie SI (
                        ( Nom + Prénom + DTNAI ) du flux == ( Nom + Prénom + DTNAI ) en BDD pour le DEM du NUMDEMRSA
                            SI ( OK )
                            ALORS ( UPDATE )
                    )
                    SINON ( REJET )
            )
            SINON
                ( INSERT )

        )
        SINON
            ( INSERT )

SINON
    ( INSERT )
```