

Logiciel Collec-Science

Installation, configuration et administration

1^{er} mars 2020 – Version 2.4.0

INRAE

Institut national de recherche pour
l'agriculture, l'alimentation et l'environnement



Site web de Collec-Science : <https://www.collec-science.org>

Éric Quinton

Document distribué sous licence CC-BY



<https://creativecommons.org/licenses/by/4.0/fr/legalcode>

Table des matières

1	Le logiciel Collec-Science	1
1.1	Historique	1
1.2	Fonctionnalités générales	2
1.3	Technologie employée	3
1.3.1	Base de données	3
1.3.2	Langage de développement et framework utilisé	3
1.3.3	Liste des composants externes utilisés	3
1.4	Sécurité	5
1.5	Licence	6
1.6	Copyright	6
2	Installer le logiciel	7
2.1	Consultez la documentation du framework !	7
2.2	Installation automatique	7
2.2.1	Mode opératoire	7
2.3	Installation manuelle	8
2.3.1	Configurer le serveur	8
2.3.2	Configurer Apache	8
2.3.3	Modules PHP nécessaires	8
2.3.4	Installer et configurer php	8
2.3.5	Configurer l'antivirus	9
2.3.6	Configurer l'hôte virtuel et SSL	9
2.3.7	Configurer Apache pour l'identification à partir d'une fédé- ration	10
2.3.8	Configurer le dossier d'installation	12
2.3.9	Droits à attribuer au serveur web	14
2.4	Configurer l'application	14
2.4.1	Connexion à la base de données	15
2.4.2	Identification des utilisateurs	15
2.4.3	Configuration de l'accès à l'annuaire LDAP	17
2.4.4	Paramètres spécifiques	18
2.4.5	Paramètres stockés en base de données	18
2.5	Créer la base de données	19
2.5.1	Créer la base de données et ajouter les extensions	19
2.5.2	Compte par défaut	19
2.5.3	Scripts de modification	19
2.6	Mise en production	20
2.7	Installer une nouvelle version	20
2.7.1	Faites une sauvegarde de la base de données	20

2.7.2	Sauvegarder le fichier contenant les paramètres de l'application	20
2.7.3	Consultez le fichier news.txt	20
2.7.4	Mise à jour de la structure de la base de données	21
2.7.5	Reconfigurer les droits d'accès au serveur web	21
2.7.6	Supprimer les dossiers inutiles	21
2.7.7	Vérifier la configuration du chiffrement	21
3	Administrer l'application	23
3.1	Gérer les droits	23
3.1.1	Principe général	23
3.1.2	Créer un nouvel utilisateur	25
3.1.3	Créer un login utilisé dans la gestion des droits	26
3.1.4	Définir les groupes d'utilisateur	26
3.1.5	Créer une application	27
3.1.6	Définir les droits utilisables dans l'application	28
3.1.7	Cas particulier des groupes et des logins issus d'un annuaire LDAP	28
3.2	Droits spécifiques de l'application COLLEC	29
3.2.1	Droits à positionner	29
3.2.2	Gestion des collections	29
3.3	Configurer les paramètres généraux	30
3.4	Créer ou modifier un modèle d'étiquettes	31
3.4.1	Définir le contenu du QRcode	32
3.4.2	Configuration du fichier XSL	33
3.5	Gestion des traces	35
4	Comment faire pour ?	37
4.1	Générer une liste d'échantillons vides	37
4.1.1	Structure du fichier CSV	37
4.1.2	Procédure d'import	38
4.1.3	Autre usage	39
4.1.4	Exemple de fichier	40
4.2	Export de données au format JSON	40
4.2.1	Description des modèles d'export	40
4.2.2	Importer un fichier JSON	41
A	Mettre en place une réplication de la base postgresql vers un autre serveur	43
A.1	Présentation	43
A.1.1	Besoins exprimés	43
A.1.2	Principe	43
A.1.3	Limitations et précautions	43
A.2	Mise à jour du serveur (version 9.3) en version 9.4 dans <i>citerne-8</i> :	44
A.3	Installation de postgresSQL sur <i>chappie</i> et mise en place des clés ssh	44
A.4	Mise en place de la réplication	44
A.4.1	Maître	44
A.4.2	Esclave	45
A.5	Informations de monitoring	45
A.6	Pour tester le failover ou gérer un interruption	46

B	Structure de la base de données	47
B.1	Description des tables	48
B.1.1	Schema col	48
B.1.2	Schema gacl	68
	Bibliographie	73

Chapitre 1

Le logiciel Collec-Science

1.1 Historique

L'unité de recherche *Écosystèmes aquatiques et changements globaux* d'IRSTEA, à Cestas (33), récolte et manipule des échantillons prélevés sur le terrain (ou plutôt, principalement dans l'eau – estuaires, lacs, rivières...), et les stocke, parfois pour des durées très longues : certaines campagnes de collecte ont eu lieu il y a plus de 40 ans.

De plus en plus, des échantillons anciens sont réanalysés (analyses génétiques, étude des ossements des oreilles ou otolithes...), au gré des questions scientifiques à traiter. Le besoin de recourir à un logiciel pour gérer ces matériels est devenu une priorité.

Dans un premier temps, quelques logiciels open-source susceptibles d'être utilisés ont été étudiés. Toutefois, leurs limites sont vite apparues : problème de pérennité, ancienneté du code, modèle de distribution parfois insatisfaisant (une licence open-source est obligatoire pour assurer la pérennité à longue échéance), résistance aux attaques informatiques, fonctionnalités insuffisantes ou inadaptées au besoin.

Dans un second temps, une étude des besoins réels a été menée. De nouvelles fonctionnalités ont été rajoutées, comme la gestion du stock de matériel utilisé sur le terrain, stocké dans un hangar.

L'unité de recherche s'intégrant au niveau régional avec d'autres organismes, des collaborations avec l'Observatoire Aquitain des Sciences de l'Univers (OASU) ou l'université de La Rochelle ont été envisagées. Des échanges productifs ont ainsi pu être mis en place, entre autres sur la gestion de l'étiquetage et le scanage des codes-barres.

Le logiciel a largement évolué suite à ces échanges, de nombreuses fonctionnalités ont été rajoutées ou modifiées pour tenir compte des besoins des partenaires potentiels.

Les délais de développement de la première version opérationnelle se sont étalés sur 9 mois, entre la définition des besoins et le développement proprement dit. La première version est parue à l'automne 2016, la version 2.0 est sortie en mai 2018.

Le code comprend environ 15800 lignes (commentaires compris), dont 7600 concernent l'affichage des pages web. Il a été écrit en PHP, les pages web sont générées en HTML et Javascript avec le composant Smarty.

1.2 Fonctionnalités générales

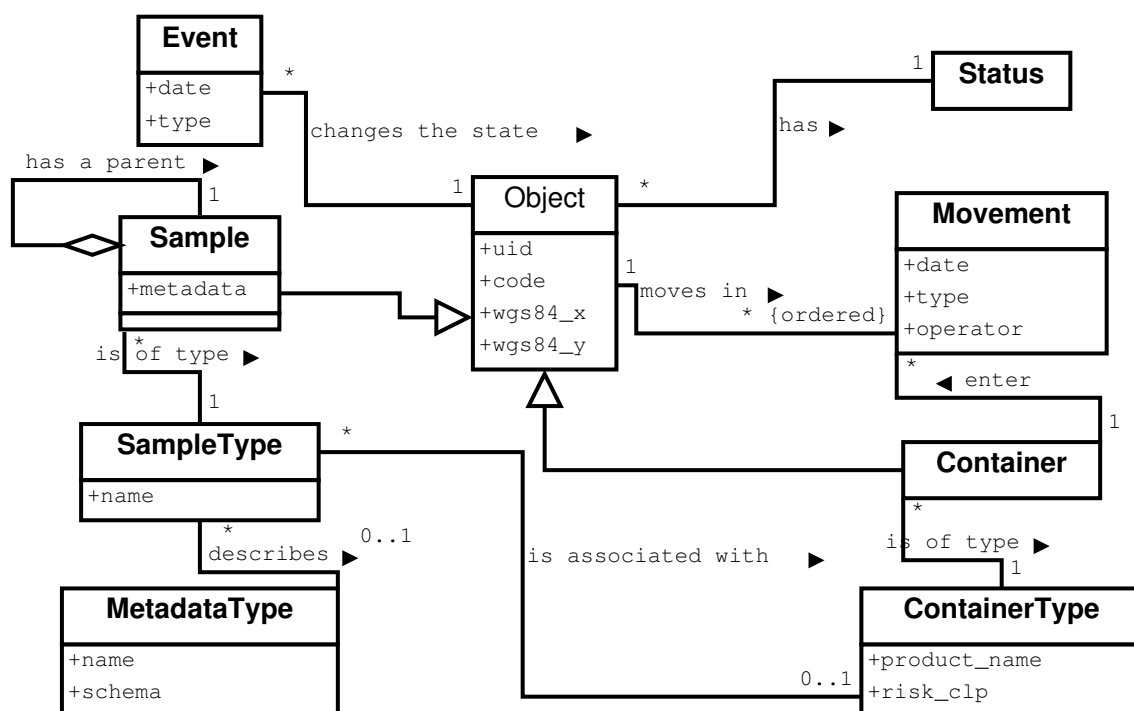


FIGURE 1.1 – Représentation objet de la base de données

Deux types d'objets sont manipulés dans l'application :

- des conteneurs (container), qui peuvent contenir des objets de tout type : d'autres conteneurs ou des échantillons. Ils peuvent être de différentes nature : site, bâtiment, salle, armoire, caisse, éprouvette... Les types de conteneurs décrivent également le produit de conservation utilisé et le risque associé (brûlure, cancérigène, etc.) ;
- des échantillons (sample), qui peuvent être associés à un type de conteneur : il y a de nombreux cas où l'échantillon lui-même se confond avec son contenant, par exemple quand le résultat d'une pêche n'est pas trié et est stocké dans un bocal.

Un échantillon ou un conteneur sont issus d'un objet unique, qui est doté :

- d'un numéro unique, l'**UID**, qui sert de référence dans le logiciel ;
- d'un identifiant métier, qui servira à le retrouver facilement (le logiciel permet également de définir d'autres identifiants).

Un objet peut subir un certain nombre d'événements, voire être réservé (fonctionnalité très simplifiée, seul le recouvrement de deux périodes de réservation est signalé).

Tout objet peut être étiqueté. Les étiquettes peuvent comprendre un code-barre 2D de type QRCode, qui pourra être lu soit à partir d'un terminal dédié (douchette), soit avec une tablette ou un smartphone, l'application disposant d'un module capable d'activer la caméra depuis le navigateur et de scanner le code-barre.

Un échantillon est obligatoirement rattaché à une collection. Seuls les personnes rattachées à celle-ci peuvent modifier les informations le concernant.

Pour mieux décrire les échantillons, il est possible de leur rattacher quelques informations « métier », appelées ici *métadonnées*. Les types de métadonnées, totalement paramétrables, sont rattachés aux types d'échantillons.

Un échantillon peut être subdivisé en d'autres échantillons. Par exemple, des otolithes (os de l'oreille) peuvent être extraits d'un poisson. Le logiciel permet de créer un nouvel échantillon à partir d'un autre, qui peut être d'un autre type le cas échéant, et qui restera associé au parent.

Enfin, dans certains cas de figures, un échantillon peut être composé de plusieurs éléments indifférenciés, par exemple plusieurs écailles de poisson prélevées et conservées ensemble. Le logiciel permet d'indiquer les prélèvements et les restitutions réalisées, et affiche le solde (théorique !) restant.

1.3 Technologie employée

1.3.1 Base de données

L'application a été conçue pour fonctionner avec Postgresql, en version 9.5. Les versions antérieures peuvent être utilisées, mais seule cette version dispose d'un type de données JSON qui permet de stocker les informations métiers.

1.3.2 Langage de développement et framework utilisé

Le logiciel a été écrit en PHP, en s'appuyant sur le framework *Prototypephp* [1], développé parallèlement par l'auteur du logiciel.

Il utilise la classe *Smarty* [2] pour gérer l'affichage des pages HTML. Celles-ci sont générées en utilisant *Jquery* [3] et divers composants associés. Le rendu général est réalisé avec *Bootstrap* [4].

Les étiquettes sont générées en utilisant FOP [5], une classe Java qui crée des fichiers PDF à partir d'un fichier XML contenant les données et un fichier de transformation au format XSL.

1.3.3 Liste des composants externes utilisés

Nom	Version	Licence	Usage	Site
PrototypePHP	branche bootstrap	LGPL	Framework	github.com/equinton/prototypephp www.smarty.net
Smarty	3.1.31	LGPL	Générateur de pages HTML	
Smarty-gettext	1.2.0	LGPL	Support multi-langues pour Smarty	
PHPCAS	1.3.5	Apache 2.0	Identification auprès d'un serveur CAS	wiki.jasig.org/display/CASC/phpCAS
PHPQRCODE	1.1.4	LGPL	Génération des qr codes	

Nom	Version	Licence	Usage	Site
Zxcvbn-PHP	0.3	MIT	Vérification de la complexité des mots de passe	

TABLE 1.1: Table des composants PHP externes utilisés dans l'application

Nom	Version	Licence	Usage	Site
Bootstrap	3.0	MIT	Présentation HTML	get.bootstrap.com
ComboBox	1.0.1	MIT	gestion des combobox	
JavaScript Cookie	2.1.4	MIT	Gestion des cookies dans le navigateur	github.com/js-cookie/
Datatables	1.10.20	MIT	Affichage des tableaux HTML	www.datatables.net
Datetime-moment		MIT	Formatage des dates dans les tableaux	datatables.net/plug-ins/sorting/datetime-moment
Moment		MIT	Composant utilisé par datetime-moment	momentjs.com
JQuery	3.3.1	≈ BSD	Commandes Javascript	jquery.com
JQuery-ui	1.12.1	≈ BSD	Commandes Javascript pour les rendus graphiques	jqueryui.com
js.cookies	0.0.4		Gestion des cookies	
leaflet	1.3.4		Affichage des cartes	
leaflet-draw	1.0.4		OpenStreetMap Dessin de polygones sur les cartes	
leaflet-mouse-position	1.2.0		récupération de la position de la souris	
leaflet-marker-cluster	1.4.1			
leaflet-tylelayer-pouchdbcached	1.0.0		Mise en cache de la cartographie	

Nom	Version	Licence	Usage	Site
pouchdb	7.1.1	MIT	moteur de mise en cache	github.com/trentrichardson/jQuery-Timepicker-Addon
Jquery-timepicker-addon			Time picker	
Magnific-popup	1.1.0	MIT	Affichage des photos	dimsemenov.com/plugins/magnific-popup/
Smartmenus	1.1.0	MIT	Génération du menu HTML	www.smartmenus.org
Openlayers	4.2.0	BSD	Affichage des cartes	openlayers.org
qcode-decoder		MIT	lecture de codes barres	cirocosta.github.io/qcode-decoder
Html5-qrcode		MIT	Lecture des QRcodes	github.com/dwa012/html5-qrcode
AlpacaJS	1.5.23	Apache 2	Génération et saisie des métadonnées	www.alpacajs.org
handlebars	4.5.3		Gestion des boutons dans AlpacaJS	
zxcvbn	4.4.2		Vérification de la complexité des mots de passe	

TABLE 1.2: Table des composants Javascript externes utilisés dans l'application

1.4 Sécurité

L'application a été conçue pour résister aux attaques dites opportunistes selon la nomenclature ASVS v3 [6] de l'OWASP [7]. Des tests d'attaque ont été réalisés en août 2016 avec le logiciel ZapProxy [8], et n'ont pas détecté de faiblesse particulière.

La gestion des droits est conçue pour :

- qu'un utilisateur, membre d'une collection, ne puisse modifier que les échantillons qui y sont rattachés ;
- que tout utilisateur disposant des droits de gestion peut procéder à une entrée ou une sortie d'un objet, quel qu'il soit ;
- que les responsables d'une collection soient les seuls à pouvoir modifier les paramètres comme les types d'échantillons ou de conteneurs, les protocoles ou les opérations rattachées.

L'analyse de sécurité a mis en exergue un besoin de ne pas perdre d'information : si un échantillon est étiqueté et rangé, et que l'information est perdue, il y a de gros risques de ne plus pouvoir l'utiliser ultérieurement. Cela impose la mise en place d'un mécanisme de réplication de la base de données, à implémenter – ou faire implémenter par des administrateurs du système – directement dans Postgresql.

1.5 Licence

Le logiciel est diffusé selon les termes de la licence GNU AFFERO GENERAL PUBLIC LICENSE version 3, en date du 19 novembre 2007 [9].

1.6 Copyright

L'application a été déposée par IRSTEA auprès de l'Agence de protection des programmes [10], sous le numéro IDDN.FR.001.470013.000.S.C.2016.000.31500

Chapitre 2

Installer le logiciel

2.1 Consultez la documentation du framework !

Le logiciel a été conçu à partir du framework *Prototypephp*. La documentation associée [11] récapitule l'ensemble des informations nécessaires pour réaliser l'installation générale (configuration du serveur, définition des droits d'accès, etc.).

De nombreux passages ont été repris ici, mais il n'est pas inutile de se référer au document d'origine.

2.2 Installation automatique

Le logiciel est livré avec un script qui installe automatiquement les paquets nécessaires, télécharge le code de l'application depuis Github, crée la base de données et prépare la configuration du serveur web Apache.

L'installateur est conçu pour fonctionner avec une distribution Debian ou Ubuntu (version LTS).

2.2.1 Mode opératoire

- installez une distribution Linux (préférentiellement, la dernière Debian stable) ;
- connectez-vous en mode *root* ;
- téléchargez le script d'installation : `deploy_new_instance.sh` avec la commande :

```
wget https://github.com/lrstea/collec/raw/master/install/
deploy_new_instance.sh
chmod +x deploy_new_instance.sh
```

- exécutez le script, qui va réaliser l'ensemble des opérations automatisables ;
- éditez ensuite le fichier `/etc/apache2/sites-available/collec-science.conf`, pour positionner correctement le dns de votre application et indiquer les informations liées au certificat https (clé privée, certificat, autorité de certification) ;
- activez le site, puis rechargez Apache :

```
a2ensite collec-science
systemctl reload apache2
```

2.3 Installation manuelle

2.3.1 Configurer le serveur

L'application est conçue pour fonctionner à partir d'une adresse unique de type : *https://monsite.com*. Le chiffrement est obligatoire (protocole https). Il n'est pas possible d'installer l'application dans un sous-dossier, par exemple : *https://monsite.com/collec-science* ne fonctionnera pas.

2.3.2 Configurer Apache

Les modules suivants doivent être activés :

```
a2enmod ssl
a2enmod headers
a2enmod rewrite
```

2.3.3 Modules PHP nécessaires

PHP doit être en version 7.2 au minimum. Il est préférable d'installer PHP depuis le site de PHP plutôt que d'utiliser les paquets fournis par la distribution.

Modules complémentaires nécessaires :

- *php-mbstring*
- *php-pgsql*
- *php7.x-xml*
- *php-xdebug* pour les phases de mise au point
- *php-curl* pour l'identification via un serveur CAS.

La génération des étiquettes nécessite les paquetages suivants :

- *php-gd*
- *fop* (qui inclut des bibliothèques java)

Le stockage et l'affichage des photos nécessite :

- *php-imagick*

2.3.4 Installer et configurer php

Voici un extrait du script d'installation automatique qui permet d'installer PHP :

```
PHPVER=7.3
PHPINIFILE="/etc/php/$PHPVER/apache2/php.ini"
apt -y install lsb-release apt-transport-https ca-certificates
DISTRIBCODE='lsb_release -sc'
DISTRIBNAME='lsb_release -si'
if [ $DISTRIBNAME == 'Ubuntu' ]
then
apt-get install software-properties-common
add-apt-repository -y ppa:ondrej/php
add-apt-repository -y ppa:ondrej/apache2
elif [ $DISTRIBNAME == 'Debian' ]
then
wget -qO https://packages.sury.org/php/apt.gpg | apt-key add -
echo "deb https://packages.sury.org/php/ $DISTRIBCODE main" | tee /etc/
apt/sources.list.d/php.list
fi
apt-get update
```

```
apt-get -y install unzip apache2 libapache2-mod-evasive libapache2-mod-
php$PHPVER php$PHPVER php$PHPVER-ldap php$PHPVER-pgsql php$PHPVER-
mbstring php$PHPVER-xml php$PHPVER-zip php$PHPVER-imagick php$PHPVER
-gd
```

De plus, après installation, la configuration du fichier `php.ini` doit être modifiée pour garantir un fonctionnement optimal du logiciel :

```
# adjust php.ini values
upload_max_filesize="=100M"
post_max_size="=50M"
max_execution_time="=120"
max_input_time="=240"
memory_limit="=1024M"
max_input_vars="10000"
for key in upload_max_filesize post_max_size max_execution_time
    max_input_time memory_limit
do
    sed -i "s/^\($key\) .*/\1 $(eval echo \${$key})/" $PHPINIFILE
done
sed -i "s/; max_input_vars = .*/max_input_vars=$max_input_vars/"
    $PHPINIFILE
```

Enfin, l'outil de gestion des images a besoin d'être reconfiguré, pour autoriser la manipulation des images :

```
# adjust imagick policy
sed -e "s/ <policy domain=\"coder\" rights=\"none\" pattern=\"PDF\"
    \/>/ <policy domain=\"coder\" rights=\"read|write\" pattern=\"PDF
    \" \/>/" /etc/ImageMagick-6/policy.xml > /tmp/policy.xml
cp /tmp/policy.xml /etc/ImageMagick-6/
```

2.3.5 Configurer l'antivirus

Les pièces téléchargées peuvent être analysées avec l'antivirus CLAMAV [12]. Dans un premier temps, Clamav doit être installé. Le plus simple est d'utiliser les paquetages de la distribution.

Suivez les instructions du document [13] pour l'installation et la vérification du bon fonctionnement.

Par défaut, le script d'installation automatique n'installe pas l'antivirus. Si vous souhaitez activer cette fonctionnalité, vous devrez la configurer vous-même.

2.3.6 Configurer l'hôte virtuel et SSL

L'application ne fonctionne qu'en mode SSL, les cookies de session n'étant pas transmis sur des liens non chiffrés. Voici un exemple de configuration à insérer dans le fichier `/etc/apache2/sites-available/default-ssl`

```
<Directory /var/www/html>
    Options FollowSymLinks MultiViews
    AllowOverride all
    Order allow,deny
    allow from all
</Directory>
SSLProtocol               all -SSLv3
```

```

SSLCipherSuite          ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-
                        CHACHA20-POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-
                        GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384
                        :DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES128
                        -SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256
                        -SHA384:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES256
                        -SHA:ECDHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA:DHE
                        -RSA-AES256-SHA256:DHE-RSA-AES256-SHA:ECDHE-ECDSA-DES-CBC3-SHA:ECDHE
                        -RSA-DES-CBC3-SHA:EDH-RSA-DES-CBC3-SHA:AES128-GCM-SHA256:AES256-GCM-
                        SHA384:AES128-SHA256:AES256-SHA256:AES128-SHA:AES256-SHA:DES-CBC3-SHA
                        :!DSS
SSLHonorCipherOrder     on
SSLCompression          off
SSLSessionTickets       off

```

(attention : pas d'espace entre *Order allow* et la virgule).

La chaîne *SSLCipherSuite* est celle qui fonctionne avec Apache 2.4.24 et openssl 1.1.0f, et est issue du configurateur mis à disposition par la fondation Mozilla [14]. Vous pouvez également consulter le document édité par l'ANSSI [15].

Activez ensuite le mode SSL dans Apache :

```

a2ensite default-ssl
service apache2 restart

```

2.3.7 Configurer Apache pour l'identification à partir d'une fédération

À partir de la version 2.4.0, Collec-Science permet d'identifier les utilisateurs à partir d'une fédération d'identités, comme la fédération française RENATER ou EDUGAIN (la fédération internationale des instituts universitaires et de recherche).

Cette identification n'est possible que pour les applications accessibles depuis Internet, et s'appuie sur l'utilisation d'un module Apache dédié : *Mellon*. Elle nécessite également de récupérer les informations techniques liées à la fédération, et d'enregistrer l'application chez le fournisseur de l'identification.

Installation du module Mellon

```
apt-get install libapache2-mod-auth-mellon
```

Si le paquet `libapache2-mod-auth-mellon` n'est pas disponible (cas rencontré avec une distribution Debian stretch), vous devrez récupérer et installer les paquets suivants (dans l'ordre) :

- `libxmlsec1`
- `libxmlsec1-openssl`
- `liblasso3`
- `libapache2-mod-auth-mellon`

Vous devrez également récupérer le fichier xml de votre *provider*, ainsi que son certificat.

Génération des fichiers de configuration de l'application

Un certificat (et sa clé privée), un fichier xml doivent être générés pour l'application. Un script est disponible dans les distributions Debian. Il est également fourni dans l'application, dans le dossier `install/apache2` (`create_metadata.sh`).

Pour générer les fichiers (remplacez *collec-science.com* par vos propres valeurs) :

```
cd /etc/apache2
mkdir mellon
cd mellon
/var/www/html/collec-science/collec/install/apache2/create_metadata.sh
https://collec-science.com https://collec-science.com/mellon
```

Le certificat (fichier .cert) et le fichier xml doivent être transmis au *provider*, pour qu'il les intègre dans sa plate-forme.

Vous devez également récupérer du *provider* sa clé publique et son certificat d'autorité racine, à mettre dans le dossier *mellon*. Il doit également vous fournir un fichier xml qui contient les adresses de toutes les entités participant à la fédération.

Configurer le site virtuel

Recopiez le fichier *install/apache2/collec-science-mellon.conf* dans le dossier */etc/apache2/sites-available*, à la place du fichier *collec-science.conf*. Éditez le fichier, et remplacez toutes les chaînes *collec.mysociety.com* par votre DNS. Vérifiez également les certificats utilisés.

Par rapport au fichier classique, le fichier *collec-science-mellon.conf* contient, dans la section *<VirtualHost *443>*, les commandes suivantes :

```
# Configuration Mellon for Renater
<location />
AuthType Mellon
MellonEnable "auth"
MellonSecureCookie On
MellonUser MAIL
MellonMergeEnvVars On
MellonSubjectConfirmationDataAddressCheck Off
MellonSPPrivateKeyFile /etc/apache2/mellon/https_collec.mysociety.
com.key
MellonSPCertFile /etc/apache2/mellon/https_collec.mysociety.com.
cert
MellonSPEntityId "https://collec.mysociety.com"
MellonSPMetadataFile "/etc/apache2/mellon/https_collec.mysociety.
com.xml"
MellonIdPMetadataFile "/etc/apache2/mellon/main-idps-renater-
metadata.xml"
MellonIdPPublicKeyFile "/etc/apache2/mellon/renater-metadata-
signing-cert-2016.pem"
MellonIdPCAFile "/etc/apache2/mellon/renater-metadata-signing-cert
-2016.pem"
MellonProbeDiscoveryTimeout 1
MellonSetEnv "MAIL" "urn:oid:0.9.2342.19200300.100.1.3"
MellonSetEnv "GIVENNAME" "urn:oid:2.5.4.42"
MellonEndpointPath /mellon
MellonSetEnvNoPrefix REMOTE_USER NAME_ID
MellonDiscoveryURL "https://discovery.renater.fr/renater/WAYF"
</location>
```

Les rubriques *MellonIdP** doivent être adaptées aux fichiers fournis par votre provider.

Une fois la configuration effectuée, redémarrez le serveur Apache :

```
systemctl restart apache2
```

Configuration du logiciel

Modifiez le fichier *param/param.inc.php*, avec les informations suivantes :

```
$ident_type = "HEADER";  
$MAIL_enabled = 1;
```

Enregistrer le site dans la fédération Renater

Pour les établissements français affiliés à la fédération Renater, vous pouvez enregistrer directement votre application dans celle-ci. Des validations seront réalisées par les contacts de la fédération dans votre établissement.

Pour réaliser l'enregistrement :

- Connectez-vous au site <https://federation.renater.fr/registry>
- cliquez sur *Ajouter un fournisseur de services*
- dans l'onglet *Description*, renseignez les champs demandés, avec notamment :
 - URL du service : `https://collec.mysociety.com` (votre DNS)
- dans l'onglet *Contacts*, ne vous déclarez pas conforme au cadre de sécurité SIRTFI, sauf si vous savez ce que c'est (il y a des contraintes organisationnelles fortes pour être conforme)
- dans l'onglet *Attributs demandés*, demandez les attributs :
 - email : identification des utilisateurs (obligatoire)
 - commonName : affichage du nom des utilisateurs (obligatoire)
- dans l'onglet *Informations techniques*, indiquez l'adresse suivante pour récupérer les données de configuration :
 - URL de vos métadonnées : `https://collec.mysociety.com/mellon/metadata`

Une fois le dossier validé, vous devrez attendre le retour de votre correspondant Renater dans votre établissement, qui doit valider votre demande.

Une fois cette première demande réalisée, vous devrez vous reconnecter au site de la fédération (<https://federation.renater.fr/registry>), et activer le rattachement à la fédération choisie (onglet *Rattachement à une fédération*). Deux fichiers seront à récupérer (commande `wget` dans le dossier `/etc/apache2/mellon`) pour récupérer d'une part le certificat, et d'autre part le fichier XML contenant l'ensemble des fournisseurs attachés à la fédération.

Ce rattachement doit également être validé par votre correspondant Renater.

Attention : une fois le rattachement validé, vous devrez attendre 24 heures pour que votre application soit disponible auprès de l'ensemble des membres de la fédération, et donc pouvoir vous connecter à l'application. Dans le cas contraire, vous obtiendrez des messages d'erreur.

2.3.8 Configurer le dossier d'installation

Le principe général est que le dossier contenant l'application contient, dans son nom, le numero de version (collec-2.0 par exemple), et un lien virtuel (collec) pointe vers celui-ci. C'est le lien qui est la cible de l'adresse web : ainsi, à chaque nouvelle version, il suffit de mettre à jour le code de l'application et de faire pointer le lien vers le nouveau dossier pour que celle-ci soit opérationnelle.

Depuis la version 2.0, des scripts sont fournis pour réaliser automatiquement les mises à jour (dans le cas d'installations mono-instances).

Cas général : une seule instance hébergée dans le serveur

Utilisez le script fourni, qui créera automatiquement les dossiers nécessaires.

Cas particulier : faire cohabiter plusieurs instances avec le même code

Il est possible d'utiliser le même code applicatif pour alimenter des bases de données différentes (ou des données stockées dans des schémas différents). Cette fonctionnalité est basée sur l'attribution d'entrées DNS différentes.

Le mécanisme est décrit dans la figure 2.1 *Schéma général d'implémentation pour utiliser le même code avec des noms d'application et des jeux de données différents*, page 13.

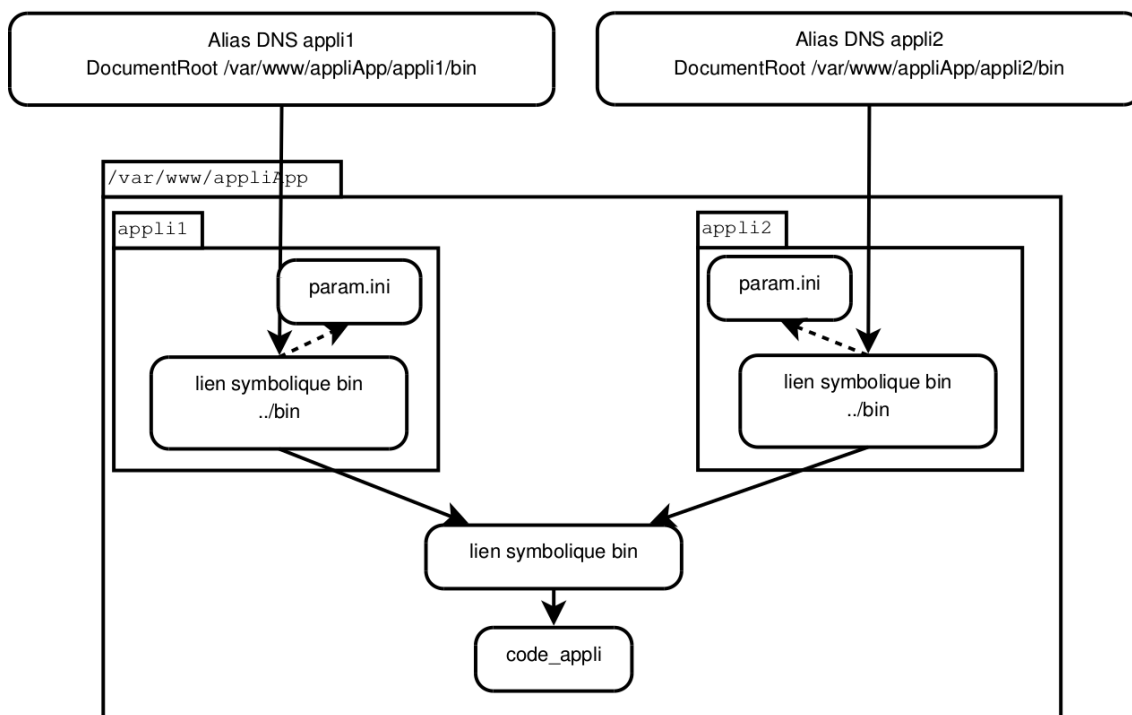


FIGURE 2.1 – Schéma général d'implémentation pour utiliser le même code avec des noms d'application et des jeux de données différents

Dans le paramétrage de l'alias DNS (en principe, dans `/etc/apache2/sites-available`), l'application pointe vers le dossier `/var/www/appliApp/appli1/bin`. `/var/www` correspond à la racine du site web, `appliApp` au dossier racine de l'application, `appli1` au dossier spécifique de l'alias DNS. Ce dossier `appli1` ne contient que deux fichiers : `param.ini`, qui contient les paramètres spécifiques, et `bin`, qui est un lien symbolique vers le dossier `../bin`.

Le dossier `../bin` (donc, dans `/var/www/appliApp`) est lui aussi un alias qui pointe vers le code réel de l'application, ici `code_appli`. Le fichier `param.inc.php` doit contenir les commandes suivantes pour que le fichier `param.ini` soit correctement chargé selon le contexte :

```
$chemin = substr($_SERVER["DOCUMENT_ROOT"],0, strpos($_SERVER["DOCUMENT_ROOT"], "/ bin"));
$paramIniFile = "$chemin/param.ini";
```

Le fichier `param.ini` sera cherché dans le dossier parent du code de l'application, c'est à dire soit dans `appli1`, soit dans `appli2` dans cet exemple. Il suffit qu'il

contienne les paramètres adéquats pour rendre l'application utilisable dans des contextes différents à partir du même code initial.

Le fichier *param.ini* est le dernier qui est traité par l'application pour récupérer les paramètres. Ceux-ci sont lus dans l'ordre suivant :

param/param.default.inc.php → param/param.inc.php → ../param.ini

param.ini contiendra les entrées spécifiques liées au DNS utilisé pour accéder à l'application, en principe tout ou partie de celles-ci :

```
APPLI_titre="Gestion des collections d'EABX"
BDD_schema=col , public , gacl
BDD_login=compte_de_connexion
BDD_passwd=mot_de_passe_de_connexion
BDD_dsn=pgsql:host=serveur;dbname=base_de_donnees;sslmode=require
GACL_aco=col
APPLI_code=proto
```

Si un libellé contient une apostrophe, la chaîne doit être insérée dans des guillemets doubles, comme ici pour la variable *APPLI_titre*.

2.3.9 Droits à attribuer au serveur web

Le serveur web doit pouvoir accéder en lecture à l'ensemble des fichiers de l'application, et en écriture à deux dossiers :

- *display/templates_c* : fichier utilisé par Smarty pour compiler les modèles de documents HTML ;
- *temp* : dossier de génération des images et des fichiers temporaires.

Deux scripts sont fournis pour attribuer les droits :

- **install/apache2/upgrade_rights.sh** : positionne les droits en utilisant les droits standards Linux (owner, group)
- **install/apache2/upgrade_rights_with_acl.sh** : positionne les droits à partir des ACL.

Les scripts doivent être lancés ainsi :

```
collec -2.0/install/apache2/upgrade_rights.sh collec -2.0
```

ou

```
collec -2.0/install/apache2/upgrade_rights_with_acl.sh collec -2.0
```

2.4 Configurer l'application

L'application est configurable par l'intermédiaire de trois fichiers :

param/param.default.inc.php → param/param.inc.php → ../param.ini

Le premier fichier contient les paramètres par défaut. Il est systématiquement fourni à chaque nouvelle version de l'application.

Le second est spécifique de l'implémentation. Il comprend notamment les informations liées à la connexion à la base de données, à la méthode d'identification, ou à la recherche des attributs dans l'annuaire LDAP.

Le troisième est destiné à offrir la possibilité d'accéder, à partir du même code applicatif, à plusieurs bases de données différentes (cf. 2.3.8 *Cas particulier : faire cohabiter plusieurs instances avec le même code*, page 13).

Voici les principaux paramètres utilisés :

2.4.1 Connexion à la base de données

Dans la pratique, deux connexions sont nécessaires : l'une pour accéder à la base des droits, l'autre aux données proprement dites. Voici les paramètres à définir :

Variable	Signification
BDD_login	compte de connexion à la base de données
BDD_passwd	mot de passe associé
BDD_dsn	adresse de la base de données sous forme normalisée
BDD_schema	schéma utilisé (plusieurs schémas peuvent être décrits, en les séparant par une virgule - fonctionnement propre à PostgreSQL)
GACL_dblogin	compte de connexion à la base de données des droits
GACL_dbpasswd	mot de passe associé
GACL_dsn	adresse normalisée
GACL_schema	schéma utilisé
GACL_aco	nom du code de l'application utilisé dans la gestion des droits

TABLE 2.1: Variables utilisées pour paramétrer les connexions

2.4.2 Identification des utilisateurs

Variable	Signification
ident_type	Type d'identification supporté : <ul style="list-style-type: none"> — BDD : uniquement en base de données — LDAP : uniquement à partir d'un annuaire LDAP — LDAP-BDD : test de connexion d'abord auprès de l'annuaire LDAP, puis en base de données en cas d'échec — CAS : identification auprès d'un serveur CAS (Common Access Service) — HEADER : identification auprès d'une fédération d'identités, comme EDUGAIN. Nécessite un paramétrage particulier du serveur Apache2 (cf. 2.3.7)
CAS_plugin	Nom du plugin utilisé pour une connexion CAS
CAS_address	Adresse du serveur CAS, sous la forme : <i>nomserveur.societe.com/cas</i> (sans préfixer avec https ://)
CAS_port = 443	port utilisé pour atteindre le serveur CAS (port https)
CAS_debug = false	true ou false, pour activer ou non l'enregistrement des fonctions de débogage. À positionner systématiquement à <i>false</i> en production

Variable	Signification
CAS_CApath = ""	Chemin d'accès au certificat de l'autorité de certification qui correspond au certificat fourni par le serveur CAS (connexion https). Si la chaîne est vide, le certificat n'est pas vérifié. Le chemin doit être renseigné en production
LDAP	tableau contenant tous les paramètres nécessaires pour une identification LDAP
privateKey	clé privée utilisée pour générer les jetons d'identification (ré-identification automatique après une première connexion)
pubKey	clé publique utilisée pour générer les jetons d'identification
tokenIdentityValidity	durée de validité, en secondes, des jetons d'identification
MAIL_enabled	Si à 1, l'envoi de mail est géré par l'application
CONNEXION_max_attemps	nombre maximum d'essais de connexion avant blocage temporaire du compte
CONNEXION_blocking_duration	durée de blocage du compte
APPLI_mailToAdminPeriod	intervalle de temps entre l'envoi d'un mail de notification de blocage de compte à un administrateur
APPLI_admin_ttl	durée de vie d'une session d'administration (temps maximum entre deux accès à une page d'administration avant réidentification)
APPLI_lostPassword	Si à 1, autorise la récupération du mot de passe perdu, par envoi d'un mail avec un lien chiffré. Nécessite également que MAIL_enabled soit positionné à 1
indent_header_vars	tableau de configuration de l'identification en mode header <ul style="list-style-type: none"> — radical : racine des libellés des variables — login : champ renvoyé contenant le login (par défaut, le mail) — mail : champ contenant le mail — cn : common name : nom et prénom — organization : nom de l'organisation d'appartenance — organizationGranted : tableau contenant la liste des organisations autorisées

TABLE 2.2: Variables utilisées pour paramétrer l'identification

Ré-identification par jeton

L'application permet de conserver l'identification plus longtemps que celle définie dans le serveur, en jouant la connexion avec un jeton d'identification chiffré. Cela évite, par exemple, de devoir se ré-identifier toutes les heures si on accède au logiciel à partir d'un terminal mobile (smartphone ou tablette, par exemple).

Les trois dernières variables permettent de configurer ce mode d'identification.

Le framework peut générer un jeton chiffré après la première identification, qui sera analysé pour savoir si l'utilisateur peut être ré-identifié automatiquement.

Pour que ce mécanisme fonctionne, il faut :

- que le paramètre *tokenIdentityValidity* ait une durée de validité supérieure à la durée de vie de la session. Il est raisonnable de ne pas fixer une durée de vie supérieure à une journée de travail (10 heures). Le cookie transmis est protégé ;
- que les clés privée et publique, utilisées pour le chiffrement du jeton, soient accessibles au serveur web (variables *privateKey* et *publicKey*).

Le jeton est chiffré avec la clé privée, ce qui lui permet d'être lu, le cas échéant, par l'application. Il contient le login et la date d'expiration.

Si l'utilisateur déclenche une déconnexion, le jeton est supprimé.

Pour plus d'informations, consultez comment fonctionne le mécanisme de ré-identification par jeton [16].

2.4.3 Configuration de l'accès à l'annuaire LDAP

Les paramètres LDAP sont stockés dans un tableau :

```
$LDAP = array(
    "address"=>"localhost",
    "port" => 389,
    "rdn" => "cn=manager,dc=example,dc=com",
    "basedn" => "ou=people,ou=example,o=societe,c=fr",
    "user_attr" => "uid",
    "v3" => true,
    "tls" => false,
    "groupSupport"=>true,
    "groupAttrib"=>"supannentiteaffectation",
    "commonNameAttrib"=>"displayname",
    "mailAttrib"=>"mail",
    'attributgroupname' => "cn",
    'attributloginname' => "memberuid",
    'basedngroup' => 'ou=example,o=societe,c=fr'
);
```

L'application peut non seulement identifier les utilisateurs auprès de l'annuaire LDAP, mais également récupérer les groupes auxquels ils appartiennent dans celui-ci.

Voici les paramètres à indiquer dans ce cas de figure (valable en principe pour tout annuaire compatible OpenLdap) :

Variable	Signification
address	adresse de l'annuaire
port	389 en mode non chiffré, 636 en mode chiffré
rdn	compte de connexion, si nécessaire
basedn	base de recherche des utilisateurs
user_attr	nom du champ contenant le login à tester
v3	toujours à <i>true</i>
tls	<i>true</i> en mode chiffré
groupSupport	true si l'application recherche les groupes d'appartenance du login dans l'annuaire
groupAttrib	Nom de l'attribut contenant la liste des groupes d'appartenance

Variable	Signification
commonNameAttrib	Nom de l'attribut contenant le nom de l'utilisateur
mailAttrib	Nom de l'attribut contenant l'adresse mail de l'utilisateur
attributgroupname	Attribut contenant le nom du groupe lors de la recherche des groupes (cn par défaut)
attributloginname	attribut contenant les membres d'un groupe
basedngroup	base de recherche des groupes

TABLE 2.3: Variables utilisées pour paramétrer l'accès à l'annuaire LDAP

2.4.4 Paramètres spécifiques

Variable	Signification
GACL_aco	nom du code de l'application utilisé dans la gestion des droits (cf. section 3.1)
APPLI_code	obsolète. Voir la section 2.4.5
APPLI_print_direct _command	Commande utilisée pour l'impression directe (depuis le serveur des étiquettes). Par défaut, <i>lpr</i> , mais <i>lp</i> peut être utilisé pour les Raspberry.
APPLI_virusScan = false	Variable qui permet d'activer le contrôle antivirus des pièces téléchargées, si Clamav est installé dans le serveur
APPLI_max_file_size = 10	Taille maxi en MB des fichiers téléchargés

TABLE 2.4: Variables spécifiques

2.4.5 Paramètres stockés en base de données

À partir de la version 1.2, certains paramètres peuvent être stockés dans la base de données, pour éviter qu'ils ne soient dépendants de la configuration du serveur.

Ces paramètres sont accessibles depuis le menu *administration*, item *Paramètres de l'application*.

Voici la liste des paramètres actuellement décrits :

Variable	Signification
APPLI_code	Code interne de l'application. Ce code est essentiel : il sera inscrit dans les codes-barres générés, pour s'assurer qu'un échantillon est bien issu de l'application (couple logiciel ↔ base de données) concernée. Il ne doit pas être modifié après avoir été attribué
APPLI_title	Titre de l'application, affiché dans le menu
mapDefaultX	Longitude de positionnement du centre de la carte par défaut
mapDefaultY	Latitude de positionnement du centre de la carte par défaut
mapDefaultZoom	facteur de zoom par défaut lors de l'affichage d'une carte

TABLE 2.5: Paramètres stockés dans la base de données

2.5 Créer la base de données

La base de données est composée de deux schémas : l'un pour stocker les informations d'identification, les droits d'accès et les traces, l'autre pour les données proprement dites.

Le schéma *public* ne devrait jamais être utilisé pour stocker l'information : réservez-le pour les composants communs, comme Postgis.

Les tables de gestion des droits peuvent être communes à plusieurs jeux / applications différentes : la variable *GACL_aco* permet de séparer la gestion des droits pour chaque application, tout en travaillant à partir des mêmes utilisateurs (répartis le cas échéant dans des groupes différents selon le jeu de données considéré).

Les scripts de création des schémas dans la base de données sont stockés dans le dossier *install/pgsql*.

2.5.1 Créer la base de données et ajouter les extensions

La base de données doit être créée avec le superutilisateur postgres. Le script *install/init_by_psql.sql* permet de réaliser les opérations suivantes :

- création du login postgresql *collec*
- création de la base de données *collec*
- ajout des extensions nécessaires (deux concernent la création des index, une pour les données géographiques, et la dernière pour implémenter les fonctions cryptographiques)
- connexion avec le login *collec* à la base de données *collec*
- exécution du script de création des schémas et des tables

Ce script peut être exécuté ainsi :

```
cd install
su postgres -c "psql -f init_by_psql.sql"
```

Si la base de données est hébergée dans un serveur différent du serveur web, il faut paramétrer auparavant Postgresql pour autoriser la connexion avec le login *collec* depuis le serveur web, en modifiant le fichier */etc/postgresql/11/main/pg_hba.conf* :

```
host collec collec adresse_serveur/32 md5
```

Le premier *collec* correspond au nom de la base de données, le second au login autorisé depuis l'adresse indiquée.

La configuration de Postgresql doit être rechargée :

```
systemctl reload postgresql
```

2.5.2 Compte par défaut

Le script crée un compte d'administration par défaut :

- login : **admin**
- mot de passe : **password**

Il devra être supprimé quand un autre compte d'administration aura été créé.

2.5.3 Scripts de modification

Lors de la livraison de nouvelles versions, il est possible que des scripts de modification soient livrés pour mettre à niveau la base de données. Ces scripts

doivent être exécutés dans tous les schémas contenant des données applicatives (pour plus de détails, consultez ci-après *Installer une nouvelle version*).

2.6 Mise en production

Une fois l'application configurée, et après avoir créé un nouveau compte d'administration :

- supprimez le compte *admin*, livré par défaut, qui ne doit pas être conservé. Sa désactivation n'est pas suffisante : si pour une raison ou pour une autre le compte est réactivé, n'importe qui pourra récupérer les droits totaux ;
- supprimez le dossier *install* qui contient les scripts de création des tables ;
- déplacez le dossier *database*, qui contient la documentation d'installation et de configuration (elle n'a pas à rester accessible depuis le site web) ;
- faites une revue des droits, pour vous assurer que tout est correctement configuré.

Vous pouvez également tester si la configuration du serveur est correcte en recourant à *ZAP*roxy [8], qui analysera la communication entre le serveur et un navigateur et identifiera les problèmes éventuels de non conformité (mauvaise réécriture des entêtes HTML suite à une mauvaise configuration du serveur Apache, par exemple).

2.7 Installer une nouvelle version

2.7.1 Faites une sauvegarde de la base de données

Il arrive fréquemment que la structure de la base de données évolue. Avant toute opération, assurez-vous de disposer d'une sauvegarde, dans un autre support.

Un programme de sauvegarde est disponible dans *install/pgsql/backup.sh*. Vous pouvez l'exécuter manuellement ainsi :

```
su postgres -c "install/pgsql/backup.sh"
```

La sauvegarde sera stockée dans **/var/lib/postgresql/backup**.

Si vous avez utilisé le script d'installation automatique, le programme est également présent dans */var/lib/postgresql*.

2.7.2 Sauvegarder le fichier contenant les paramètres de l'application

Le fichier *param/param.inc.php* contient vos paramétrages spécifiques. Lors de l'installation d'une nouvelle version, il va être supprimé.

Faites-en une copie, et remettez-le en place après avoir installé la nouvelle version.

2.7.3 Consultez le fichier *news.txt*

Le fichier *param/news.txt* contient la description des modifications apportées au logiciel. Il précise notamment si une mise à jour de la base de données doit être appliquée.

2.7.4 Mise à jour de la structure de la base de données

Le dossier *install/pgsql* contient les scripts de création et de mise à jour de la base de données. Les scripts de mise à jour sont nommés ainsi :

```
col_alter_versionAnterieure-versionMiseAJour.sql
```

versionAnterieure correspond à la version la plus ancienne qui doit être mise à jour, *versionMiseAJour* la version cible. Par exemple :

```
col_alter_1.2-1.2.3.sql
```

indique que toutes les versions entre 1.2 et 1.2.3 doivent être mises à jour avec le script indiqué. Si vous avez « sauté » certaines versions du logiciel, il est possible que plusieurs scripts doivent être appliqués.

La mise à jour doit être appliquée dans tous les schémas contenant des données, notamment dans le cas où le même logiciel est utilisé pour gérer plusieurs jeux de données.

Avant d'exécuter les scripts, vérifiez leur contenu, et notamment le nom des schémas.

Ne relancez jamais l'exécution d'un script.

2.7.5 Reconfigurer les droits d'accès au serveur web

Après installation de la nouvelle version du code, n'oubliez-pas de reconfigurer les accès en lecture pour le compte utilisé pour faire fonctionner le serveur web, et en écriture pour les dossiers *temp* et *display/templates_c* (cf. 2.3.9 *Droits à attribuer au serveur web*, page 14).

2.7.6 Supprimer les dossiers inutiles

Une fois la mise en production validée, supprimez les dossiers *install* et *database*, et faites une revue des droits pour vous assurer qu'il n'y a pas eu de modification intempestive ou que la configuration est toujours correcte.

2.7.7 Vérifier la configuration du chiffrement

Avec un navigateur récent, ou en testant le site (s'il est accessible depuis internet) à partir de [SSLLABS](https://ssllabs.com), vérifiez que l'application soit correctement configurée, notamment au niveau du serveur Apache.

Chapitre 3

Administrer l'application

3.1 Gérer les droits

Depuis la version 1.1, les scripts de création des bases de données intègrent la génération initiale des groupes et des droits associés, ceci afin de faciliter la phase de mise en route.

Toutefois, vous devrez créer des groupes d'utilisateurs correspondant à vos projets, et modifier ensuite les projets pour donner les droits adéquats aux groupes créés (cf. 3.2.2, page 29).

3.1.1 Principe général

Les droits sont gérés selon le principe initialement utilisé dans la bibliothèque PHPGACL [17], aujourd'hui obsolète.

Les logins sont déclarés dans des groupes organisés de manière hiérarchique : un groupe hérite des droits attribués à ses parents.

Les droits utilisés dans le logiciel sont associés à des groupes. Il est possible d'attribuer plusieurs droits à un même groupe, et un droit peut être détenu par des groupes différents.

Si le paramètre `$LDAP["groupSupport"]` est positionné à `true`, les groupes dont fait partie le compte LDAP sont également récupérés. Si ces groupes se voient attribués des droits, les comptes associés les récupéreront automatiquement.

Voici le schéma des tables utilisées pour gérer les droits :

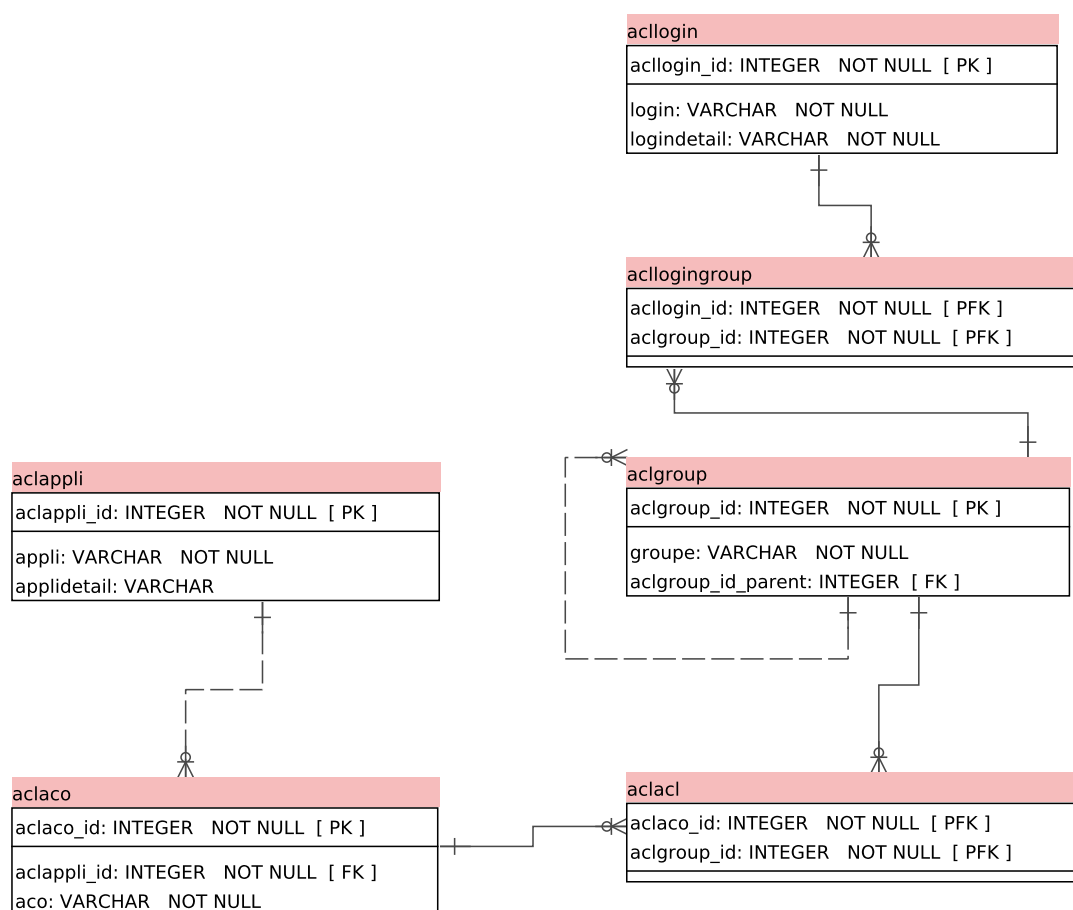


FIGURE 3.1 – Schéma des tables utilisées pour gérer les droits

Voici la description des tables :

aclogin : liste des logins utilisés. Si un compte est créé dans la base locale d'identification, un enregistrement est également créé dans cette table. Pour les identifications LDAP ou CAS, ils doivent être identiques. Si seuls les groupes LDAP sont utilisés pour un compte, il n'a pas besoin d'être décrit ici ;

aclappli : liste des applications gérées. Il est possible de gérer, à partir de la même base de données, plusieurs ensembles de droits, qui utilisent les mêmes logins.

aclaco : liste des droits déclarés dans l'application ;

aclgroup : liste des groupes contenant les logins, et qui détiennent les droits. Un groupe peut hériter d'un autre groupe. Les droits associés au groupe parent sont également attribués au groupe hérité ;

acloggingroup : table permettant de déclarer les logins associés à un groupe ;

aclacl : table décrivant les droits détenus par un groupe.

Le module d'administration permet de saisir toutes ces informations. Il faut que l'utilisateur dispose du droit *admin*, c'est à dire qu'il fasse partie du groupe *admin* (configuration par défaut à l'initialisation de la base des droits) pour pouvoir accéder à ces fonctions.

3.1.2 Créer un nouvel utilisateur

Les utilisateurs peuvent être issus soit de l'annuaire LDAP, soit de la base interne. Pour créer un nouvel utilisateur dans la base locale :

- Administration → Liste des comptes
- Nouveau login
- renseignez au minimum le login.

The image shows a web form for creating a new user. It has a light blue background. The form fields are: 'Login *' with the value 'test'; 'Nom'; 'Prénom'; 'Mèl'; 'Date' with the value '28/06/2016'; 'Mot de passe' and 'Répétez le mot de passe', both with a key icon on the right; a section for 'Générez un mot de passe aléatoire' with a green 'Générez' button and an empty text box; and an 'actif' section with radio buttons for 'oui' (selected) and 'non'. At the bottom are two buttons: 'Valider' (blue) and 'Supprimer' (red).

FIGURE 3.2 – Écran de saisie d'un login de connexion

Pour créer le mot de passe, vous pouvez cliquer sur le bouton *Générez*, qui en générera un automatiquement. Envoyez-le par mèl à son destinataire (par *copier-coller*), en lui demandant de le modifier à la première connexion (icône en forme de clé, dans le bandeau, en haut à droite).

Les mots de passe doivent respecter les règles suivantes :

- ils doivent avoir une longueur minimale de 8 caractères ;
- ils doivent comprendre trois types de caractères différents parmi les minuscules, majuscules, chiffres et caractères de ponctuation ;
- ils ne peuvent pas être réutilisés pour le même login ;
- les mots de passe n'expirent pas.

Les mots de passe sont stockés sous forme d'empreinte, calculée en rajoutant un sel¹ et encodés en SHA256 : ils ne peuvent pas être retrouvés en cas de perte.

1. chaîne de caractère rajoutée au mot de passe – en général le login ou un identifiant – qui permet d'éviter que deux mots de passe identiques, associés à deux logins différents, aient la même empreinte

L'application n'intègre pas de module permettant de régénérer automatiquement un mot de passe en cas de perte : c'est au responsable applicatif d'en fournir un nouveau.

La création d'un compte entraîne la création d'une entrée identique dans la table des *aclogin*, utilisée pour attribuer les droits.

Pour désactiver temporairement un compte, sélectionnez *non* dans la zone *actif*. Si le compte ne doit plus être utilisé, supprimez-le.

Attention : si le compte disposait des droits d'administration, assurez-vous que vous avez toujours un compte disposant des mêmes droits avant la suppression.

3.1.3 Créer un login utilisé dans la gestion des droits

Indépendamment du compte de connexion, qui peut être soit issu de la base interne, soit récupéré auprès d'un annuaire LDAP ou d'un serveur CAS, l'application a besoin de connaître les utilisateurs pour pouvoir leur attribuer des droits.

À partir du menu, choisissez *Administration* → *ACL - logins*.

Vous pouvez modifier un login existant ou en créer un nouveau. Dans ce cas, vous devrez indiquer au minimum le login utilisé (identique à celui qui est employé pour la connexion à l'application : base de données interne, annuaire LDAP, serveur CAS).

Modification d'un login (module de gestion des droits)

[Retour à la liste des logins](#)

Nom de l'utilisateur * :	<input type="text" value="test"/>
Login utilisé * :	<input type="text" value="test"/>
<input type="button" value="Valider"/> <input type="button" value="Supprimer"/>	

*Donnée obligatoire

Droits attribués

consult

param

FIGURE 3.3 – Écran de modification d'un login dans le module de gestion des droits

Sous l'écran de saisie figurent la liste des droits attribués à un login (en modification, le calcul n'est réalisé qu'à l'affichage de la page).

3.1.4 Définir les groupes d'utilisateur

Les groupes d'utilisateurs sont gérés selon un mécanisme d'héritage. Un groupe de haut niveau hérite des groupes précédents : si des droits ont été attribués à un groupe de niveau inférieur, un login associé à un groupe de niveau supérieur les récupère également.

Pour définir les groupes, dans le menu, choisissez *Administration* → *ACL - groupes de logins*.

Nouveau groupe racine...








Nom du groupe	Nombre de logins déclarés	Rajouter un groupe fils
admin	2	
consult		
EABX		
aloston		
gestion		
projet		
param	1	

FIGURE 3.4 – Liste des groupes de logins

Ainsi, le login déclaré dans le groupe *param* récupérera les droits attribués aux groupes *projet*, *gestion* et *consult*.

Pour créer un groupe, deux possibilités :

- soit le groupe est à la base d'une nouvelle branche : utilisez alors *Nouveau groupe racine...* ;
- soit le groupe hérite d'un autre groupe : cliquez sur le signe + (*Rajouter un groupe fils*).

Vous pouvez indiquer les logins qui sont rattachés à ce groupe.

3.1.5 Créer une application

Le moteur utilisé pour faire fonctionner le logiciel COLLEC permet de gérer des droits différents pour des jeux de données différents, à partir du même code applicatif. Chaque couple *logiciel* ↔ *base de données* constitue donc une *application*, au sens de la gestion des droits.

Il est ainsi possible, à partir de la même base de données, de définir des droits différents selon les jeux de données utilisés (un jeu de données correspond à un schéma de base de données comprenant l'intégralité des tables applicatives).

À partir du menu, choisissez *Administration* → *ACL - droits* :

Modifier 	Nom de l'application 	Description 
	col	Gestion des collections d'échantillons

FIGURE 3.5 – Liste des applications déclarées

Pour créer une nouvelle application, choisissez *Nouvelle application....*

*Donnée obligatoire

FIGURE 3.6 – Écran de saisie d'une application

Le nom de l'application doit impérativement correspondre à la valeur *\$GACL_appli* dans les fichiers de paramètres : c'est ce qui permet au framework de savoir quels droits appliquer.

3.1.6 Définir les droits utilisables dans l'application

À partir de la liste des applications, cliquez sur le nom de celle pour laquelle vous voulez définir les droits utilisables. À partir de la liste, sélectionnez *Nouveau droit...*

FIGURE 3.7 – Écran de saisie des droits associés à une application

Le nom du droit doit être celui défini dans le corps de l'application (les droits sont positionnés dans les fichiers *param/actions.xml*, qui contient la liste des modules utilisables, et *param/menu.xml*, qui sert à générer le menu – cf. table 3.1 *Droits à positionner*, page 29).

Indiquez les groupes d'utilisateurs qui seront associés au droit courant.

3.1.7 Cas particulier des groupes et des logins issus d'un annuaire LDAP

Si vous avez paramétré l'application pour qu'elle s'appuie sur un annuaire LDAP pour gérer l'affectation des utilisateurs dans les groupes, vous n'êtes pas obligés de les déclarer explicitement dans le module de gestion des droits.

Droits attribués à un groupe LDAP

Tous les utilisateurs d'un groupe héritent d'un droit dans l'application.

- définissez le nom du groupe (en respectant la casse) dans le tableau des groupes d'utilisateurs (par exemple, EABX) ;
- sélectionnez le nom de ce groupe dans les droits utilisables ;
- tous les utilisateurs de l'annuaire LDAP récupéreront automatiquement les droits attribués à ce groupe.

Droits attribués à un utilisateur particulier de l'annuaire LDAP

Un utilisateur s'identifie auprès de l'annuaire LDAP, mais dispose de droits particuliers.

- créez son login dans la gestion des droits ;
- rajoutez-le dans le groupe d'utilisateurs adéquat.

3.2 Droits spécifiques de l'application COLLEC

3.2.1 Droits à positionner

Voici les droits nécessaires pour faire fonctionner correctement l'application :

Droit	Usage
admin	Gestion des utilisateurs et des droits
param	Définition des tables de paramètres généraux, gestion des collections
collection	rajout des types d'échantillons ou de conteneurs, import de masse
import	permet de réaliser des importations de masse
gestion	Ajout d'un échantillon pour les projets autorisés, entrée/sortie. Droit attribué par défaut si l'utilisateur fait partie d'au moins un projet
consult	Consultation des informations, sans possibilité de modification. Le droit de consultation doit être indiqué volontairement

TABLE 3.1: Liste des droits utilisés

Ces droits doivent être définis pour chaque application (couple *logiciel* ↔ *base de données*) gérée par la base de gestion des droits.

3.2.2 Gestion des collections

Les échantillons étant obligatoirement rattachés à une collection, vous devrez en créer au minimum une à partir du menu de paramétrage. Un utilisateur avec les droits de gestion ne peut modifier que les échantillons pour lesquels il est autorisé (les collections qui sont rattachées au(x) groupe(s) dont il fait partie).

Voici le principe de gestion des droits pour les collections :

- Dans *Administration* > *ACL - Groupes de logins*, déclarez les groupes adéquats. En cas d'utilisation des groupes LDAP, les saisir avec la même casse que dans l'annuaire (EABX p. e.).

Il est possible de définir une hiérarchie des groupes, quelle que soit l'origine de l'affectation (base de données ou annuaire Ldap). Dans le cas où l'annuaire Ldap n'est pas utilisé pour gérer les groupes, renseignez les logins en face des groupes dans le même écran ;

- Dans les collections, sélectionnez les groupes autorisés (*cf. 3.7 Écran de saisie des droits associés à une application, page 28*) ;
- les utilisateurs faisant partie des groupes autorisés disposeront des droits de *gestion* pour la collection considérée.

3.3 Configurer les paramètres généraux

L'ensemble des paramètres sont accessibles à partir du menu *Paramètres*.

Par défaut, tous les utilisateurs qui disposent du droit de consultation peuvent visualiser les paramètres. La modification n'est possible que pour ceux qui disposent des droits suivants :

Nom	Description	Droit nécessaire
Projets	Liste des projets et droits associés	admin
Protocoles	Protocoles de prélèvement des échantillons	collection
Opérations	Opérations rattachées aux protocoles	collection
Type d'événement	Événements survenant aux objets	param
Familles de conteneurs	Mécanisme pour retrouver les conteneurs selon leur nature (pièce, caisse...)	param, projet
Conditions de stockage	Mécanisme de conservation (lyophilisation, p. e.)	param, collection
Motifs de déstockage	Raisons invoquées pour sortir un objet du stock	param, collection
Types de conteneurs	Modèles de conteneurs (porteurs des étiquettes, entre autres)	param, collection
Statut des objets	Liste des statuts que peut prendre un objet	param
Type d'échantillon	Modèles des échantillons (rattachables à un type de conteneur)	param, collection
Sous-échantillonnage	Pour les échantillons composés d'éléments non différenciables, unité utilisée pour réaliser le sous-échantillonnage (nombre, volume...)	param
Étiquettes	Modèles des étiquettes imprimables	param
Types d'identifiants	Types d'identifiants complémentaires des objets	param

TABLE 3.2: Liste des paramètres et droits de modification associés

3.4 Créer ou modifier un modèle d'étiquettes

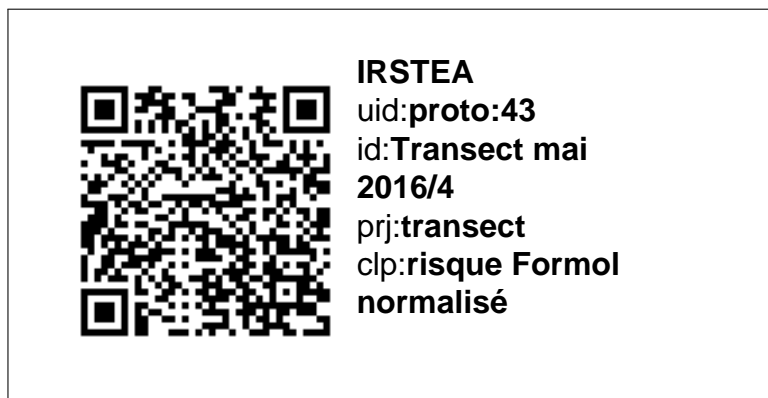


FIGURE 3.8 – Exemple d'étiquette

Les étiquettes sont créées en recourant au logiciel FOP [5], écrit en Java. Voici les opérations réalisées par l'application pour générer les étiquettes :

- pour chaque objet concerné (des containers ou des échantillons associés à un type de container, et si le type de container est rattaché à un modèle d'étiquettes), une image du QRcode est générée dans le dossier *temp* ;
- dans le dossier *temp*, un fichier au format XML est généré, contenant les informations à imprimer sur l'étiquette ;
- un fichier au format XSL, qui contient les ordres de création de l'étiquette, est également créé dans le même dossier. Le contenu de ce fichier est issu d'un enregistrement provenant de la table *label* ;
- le programme PHP fait appel à FOP pour générer, à partir du fichier XML et en utilisant le fichier XSL, un fichier PDF. Une page du fichier correspond à une étiquette (mécanisme utilisé par les imprimantes à étiquettes pour les séparer).

La configuration du modèle d'étiquettes revient à définir à la fois le contenu des informations qui seront insérées dans le QRCODE et la forme que prendra l'étiquette, c'est à dire les informations qui seront imprimées, le format, etc. Cette forme reprend la syntaxe XSL comprise par FOP.

Nom de l'étiquette* :

Bocaux transect

Transformation XSL* :

```

<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:fo="http://www.w3.org/1999/XSL/Format">
  <xsl:output method="xml" indent="yes"/>
  <xsl:template match="objects">
    <fo:root>
      <fo:layout-master-set>
        <fo:simple-page-master master-name="label"
          page-height="5cm" page-width="10cm" margin-left="0.5cm" margin-top="0.5cm" margin-bottom="0cm" margin-right="0.5cm">
          <fo:region-body/>
        </fo:simple-page-master>
      </fo:layout-master-set>

      <fo:page-sequence master-reference="label">
        <fo:flow flow-name="xsl-region-body">
          <fo:block>
            <xsl:apply-templates select="object" />
          </fo:block>
        </fo:flow>
      </fo:page-sequence>
    </fo:root>
  </xsl:template>
</xsl:stylesheet>

```

Champs à insérer dans le QRCode (séparés par une virgule, sans espace)* :

uid,id,clp,db,tag,cab,prj

Valider

Supprimer

Champs utilisables dans le QRcode et dans le texte de l'étiquette :

- uid (obligatoire) : identifiant unique
- db (obligatoire) : code de la base de données (utilisé pour éviter de mélanger les échantillons entre plusieurs bases)
- id : identifiant général
- prj : code du projet
- clp : code de risque
- pn : nom du protocole
- et tous les codes d'identifiants secondaires - cf. paramètres > Types d'identifiants

FIGURE 3.9 – Écran de saisie d'un modèle d'étiquette

3.4.1 Définir le contenu du QRcode

Le QRcode est un format de code barre normalisé en deux dimensions, qui permet de stocker jusqu'à 2000 caractères en 8 bits.

Le principe retenu dans l'application est de stocker l'information au format JSON. Pour limiter la taille du code barre, les noms des balises doivent être les plus petites possibles. Voici les balises obligatoires à insérer systématiquement dans une étiquette :

Nom	Description
uid	Identifiant unique de l'objet dans la base de données
db	Identifiant de la base de données. C'est la valeur du paramètre <i>APPLI_code</i> (cf. 2.4.4 Paramètres spécifiques, page 18)

TABLE 3.3: Liste des balises à insérer obligatoirement dans les QRcodes

D'autres informations peuvent être également insérées :

Nom	Description
id	Identifiant métier principal (champ <i>identifiant ou nom</i> , en saisie)
prj	Code de la collection (pour les échantillons)
clp	Code du risque associé au conteneur, en raison du produit de conservation utilisé
pn	Nom du protocole de collecte des échantillons
autres codes	tous les codes d'identification secondaires définis dans la table de paramètres <i>Types d'identifiants</i> (cf. 3.3 Configurer les paramètres généraux, page 30).
les champs utilisés dans les métadonnées	les codes des champs utilisés dans la description des métadonnées. Un modèle d'étiquette ne peut être associé qu'à un type de métadonnées

TABLE 3.4: Liste des balises facultatives insérables dans les QRcodes

3.4.2 Configuration du fichier XSL

La syntaxe particulière du fichier XSL ne doit être modifiée qu'en conservant la version initiale (recopie dans un bloc-notes, par exemple), pour éviter de perdre une configuration opérationnelle suite à un mauvais paramétrage.

Voici la description du contenu du fichier et les zones modifiables.

Entête du fichier

Elle permet de modifier la taille de l'étiquette (largeur et hauteur maximale). Vous ne devriez changer que les attributs *page-height* et *page-width*. Pour les marges (attributs *margin-*), soyez prudents et vérifiez notamment que les QR-codes ne soient pas rognés à cause de marges insuffisantes.

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:fo="http://www.w3.org/1999/XSL/Format">
  <xsl:output method="xml" indent="yes"/>
  <xsl:template match="objects">
    <fo:root>
      <fo:layout-master-set>
        <fo:simple-page-master master-name="label"
          page-height="5cm" page-width="10cm"
          margin-left="0.5cm"
          margin-top="0.5cm"
          margin-bottom="0cm"
          margin-right="0.5cm">
          <fo:region-body/>
        </fo:simple-page-master>
      </fo:layout-master-set>

      <fo:page-sequence master-reference="label">
        <fo:flow flow-name="xsl-region-body">
          <fo:block>
            <xsl:apply-templates select="object" />
          </fo:block>

        </fo:flow>
      </fo:page-sequence>
    </fo:root>
  </xsl:template>
</xsl:stylesheet>
```

```

    </fo:root>
  </xsl:template>
<xsl:template match="object">

```

Format de l'étiquette

Le contenu de l'étiquette est décrit sous la forme d'un tableau (balises *fo:table*). La première colonne contient le QRCode, la seconde le texte associé.

Ici, deux colonnes de taille identique (4 cm chacune) sont définies.

```

<fo:table table-layout="fixed" border-collapse="collapse"
border-style="none" width="8cm"
keep-together.within-page="always">
  <fo:table-column column-width="4cm"/>
  <fo:table-column column-width="4cm" />
<fo:table-body border-style="none" >

```

Les cellules (*table-cell*) sont insérées dans une ligne (*table-row*) :

```

  <fo:table-row>

```

Insertion du QRcode

Le QRcode est inséré dans un bloc. Les seules informations modifiables sont celles concernant la hauteur (attribut *height* et la largeur (attribut *content-width*)). Veillez à ce que la hauteur et la largeur soient identiques, et ne modifiez pas les autres informations.

```

    <fo:table-cell>
      <fo:block>
        <fo:external-graphic>
<xsl:attribute name="src">
  <xsl:value-of select="concat(uid, '.png') " />
</xsl:attribute>
<xsl:attribute name="content-height">
scale-to-fit
</xsl:attribute>
<xsl:attribute name="height">4cm</xsl:attribute>
  <xsl:attribute name="content-width">4cm</xsl:attribute>
  <xsl:attribute name="scaling">uniform</xsl:attribute>
</fo:external-graphic>
      </fo:block>
    </fo:table-cell>

```

Contenu textuel

Les autres informations sont affichées dans des blocs, avec une ligne par catégorie d'information.

L'étiquette commence ici par indiquer l'établissement (ici, IRSTEA), écrit en gras.

```

  <fo:table-cell>
    <fo:block>
      <fo:inline font-weight="bold">
        IRSTEA
      </fo:inline>
    </fo:block>

```


Chaque information est affichée dans un bloc, comprenant un titre (par exemple, *uid*), associé à une ou plusieurs valeurs. Ainsi, la première ligne affiche sur la même ligne, et en gras (attribut *font-weight="bold"*), le code de la base de données (*<xsl:value-of select="db"/>*) et l'UID de l'objet (*<xsl:value-of select="uid"/>*).

```
<fo:block>uid:
<fo:inline font-weight="bold">
<xsl:value-of select="db"/>:
<xsl:value-of select="uid"/></fo:inline>
</fo:block>
<fo:block>id:
<fo:inline font-weight="bold">
<xsl:value-of select="id"/></fo:inline>
</fo:block>
<fo:block>prj:
<fo:inline font-weight="bold">
<xsl:value-of select="prj"/></fo:inline>
</fo:block>
<fo:block>clp:
<fo:inline font-weight="bold">
<xsl:value-of select="clp"/></fo:inline>
</fo:block>
```

Fin de l'étiquette

Une fois toutes les informations affichées, le tableau est fermé, et un saut de page est généré systématiquement :

```
</fo:table-cell>
</fo:table-row>
</fo:table-body>
</fo:table>
<fo:block page-break-after="always"/>
```

Enfin, le fichier XSL est correctement fermé :

```
</xsl:template>
</xsl:stylesheet>
```

Il est possible de créer des étiquettes avec des formats différents, par exemple en créant plusieurs lignes. Pensez à fermer vos balises, et qu'elles soient correctement imbriquées, pour éviter tout souci.

Pour aller plus loin dans la mise en page, consultez la documentation du projet FOP.

3.5 Gestion des traces

Tous les appels lancés par les utilisateurs vers les modules de l'application sont enregistrés dans la table *gacl.log*, qui ne doit être accessible qu'aux personnes dûment autorisées. Les traces sont supprimées au bout d'un an (script de nettoyage exécuté lors de la connexion d'un utilisateur).

Voici un exemple de trace générée :

log_id	login	nom_module	log_date	commentaire
	ipaddress			
523437	eric.quinton	col-Sample-write	2016-10-25 14:57:01	
	16	::1		

523438	eric.quinton ok ::1	col-sampleDisplay	2016-10-25 14:57:01	
523436	eric.quinton ::1	col-sampleWrite	2016-10-25 14:57:00	ok
523435	eric.quinton ok ::1	col-sampleChange	2016-10-25 14:56:58	
523434	eric.quinton ok ::1	col-sampleDisplay	2016-10-25 14:56:55	
523433	eric.quinton ::1	col-sampleList	2016-10-25 14:56:52	ok
523431	eric.quinton ::1	col-default	2016-10-25 14:53:05	ok
523430	eric.quinton ::1	col-connexion	2016-10-25 14:53:04	ldap-ok
523429	unknown	col-connexion	2016-10-25 14:52:57	ok ::1

La colonne *commentaire*, pour la ligne 523437, contient l'identifiant modifié (l'enregistrement 16 a été traité par le module sampleWrite – Sample (majuscule du S) correspond au nom de la classe qui a été utilisée pour réaliser l'écriture vers la base de données). L'adresse IP est théoriquement celle de l'utilisateur (ici, connexion locale), y compris en prenant en compte le passage par un serveur Reverse-proxy².

Parallèlement, les messages d'erreur sont envoyés au processus Linux SYS-LOG, qui enregistre les traces dans le fichier */var/log/apache2/error.log*.

2. serveur mis en entrée du réseau privé, qui permet de masquer les adresses internes et de contrôler les accès depuis Internet

Chapitre 4

Comment faire pour ?

4.1 Générer une liste d'échantillons vides

Objectif : préparer des bocaux d'échantillons avant de partir en campagne de collecte. Ces bocaux doivent être étiquetés.

Le logiciel propose une procédure d'import de masse, qui permet de répondre à cette question.

Voici la méthode à utiliser :

- générez un fichier au format CSV (créé par exemple à partir de LibreOffice OpenDataSheet – ODS), qui comprend une ligne par échantillon ;
- lancez la procédure d'import : le programme vous indiquera les UID générés ;
- recherchez les UID générés, et déclenchez l'impression des étiquettes.

4.1.1 Structure du fichier CSV

Toute opération d'import présente des risques : il est difficile de revenir en arrière une fois celle-ci terminée. Pour les limiter, le logiciel va procéder en deux étapes. D'abord, la structure du fichier va être analysée, et la cohérence des informations indiquées vérifiée. Ensuite, si aucune anomalie n'est détectée, l'import pourra être déclenché.

La première ligne du fichier doit comporter le nom des colonnes. Leur nom est normalisé et ne doit en aucun cas être modifié. Si une colonne n'existe pas, l'import du fichier sera rejeté.

Les identifiants numériques (*project_id* par exemple) doivent être recherchés dans les tables de paramètres de l'application.

Voici la liste des colonnes utilisables :

Colonne	Description	Obligatoire
sample_identifieur	identifiant métier de l'échantillon	X
collection_id	identifiant de la collection de rattachement	X
sample_type_id	identifiant du type d'échantillon	X
sample_status_id	identifiant du statut à attribuer	X
sampling_place_id	le numéro informatique de l'endroit où l'échantillon a été prélevé	
wgs84_x	longitude GPS en WGS84 (degrés décimaux)	

Colonne	Description	Obligatoire
wgs84_y	latitude GPS en WGS84 (degrés décimaux)	
sampling_date	date de création/échantillonnage de l'échantillon, au format dd/mm/yyyy	
expiration_date	date d'expiration de l'échantillon, au format dd/mm/yyyy	
sample_location	emplacement de rangement de l'échantillon dans le container (texte libre)	
sample_column	n° de la colonne de stockage dans le container	
sample_line	n° de la ligne de stockage dans le container	
sample_multiple_value	le nombre total de sous-échantillons (ou le volume total, ou le pourcentage...) contenu dans l'échantillon si le type d'échantillons utilisé le permet (valeur numérique, séparateur décimal : point)	
sample_parent_uid	UID du parent (création d'échantillons rattachés)	
sample_metadata_json	métadonnées rattachées à l'échantillon (au format json, p. e. : {"taxon" : "Alosa alosa"})	
container_identifiant	identifiant du container	X
container_type_id	identifiant du type de container	X
container_status_id	identifiant du statut à attribuer au container	
container_column	n° de la colonne de stockage dans le container parent	
container_line	n° de la ligne de stockage dans le container parent	
container_parent_uid	UID du container dans lequel le container courant est rangé	
identifiants complémentaires	une colonne par code supplémentaire (menu <i>Paramètres</i> → <i>Types d'identifiants</i>)	

TABLE 4.1: Liste des colonnes utilisables lors d'un import de masse

Les champs obligatoires ne le sont que si l'identifiant de l'objet considéré – échantillon ou container – a été renseigné. Une ligne doit contenir au minimum soit un numéro d'échantillon, soit un numéro de container.

4.1.2 Procédure d'import

À partir du menu, choisissez *Objet* → *import de masse*. Seuls les utilisateurs qui disposent du droit *projet* ou *import* pourront réaliser l'opération.

Nom du fichier à importer (CSV)* : Choisissez un fichier Aucun fichier choisi

Séparateur utilisé :

Encodage du fichier :

FIGURE 4.1 – Sélection du fichier pour un import de masse

Sélectionnez le fichier à importer, vérifiez le séparateur utilisé. Préférez, si possible, les données au format UTF-8.

L'import sera réalisé ainsi :

1. si `sample_identifier` est renseigné : création de l'échantillon
2. si `container_identifier` est renseigné : création du container
3. si `container_identifier` et `container_parent_uid` sont renseignés : création du mouvement d'entrée du container
4. si l'échantillon et le container ont été créés, création du mouvement d'entrée de l'échantillon dans le container
5. si l'échantillon est créé, que `container_parent_uid` est renseigné, et que `container_identifier` n'est pas rempli, création du mouvement d'entrée de l'échantillon dans le container indiqué

Si des anomalies sont détectées lors du contrôle, un tableau récapitulant les problèmes rencontrés sera affiché, ressemblant à ceci :

N° de ligne	Anomalie(s) détectée(s)
11	Le numéro du projet indiqué n'est pas reconnu ou autorisé. Le type de container n'est pas connu.
12	Le statut du container n'est pas connu. L'UID du conteneur parent n'existe pas.
13	Aucun échantillon ou container n'est décrit (pas d'identifiant pour l'un ou pour l'autre).

FIGURE 4.2 – Exemples d'anomalies détectées lors du contrôle de l'import

Si les contrôles se sont bien déroulés, le programme proposera alors de déclencher l'import, et affichera en retour les valeurs *mini* et *maxi* des UID générées.

4.1.3 Autre usage

Cette fonctionnalité peut également être utilisée pour déclencher l'import de listes d'échantillons pré-existants, et de créer automatiquement les mouvements adéquats pour les ranger dans leurs containers de stockage.

4.1.4 Exemple de fichier

1	sample_identifiant	project_id	sample_type_id	sample_date	sample_range	container_identifiant	container_type_id	container_status_id	container_parent_uid	container_range
2	TESTOTOLITHE1	1	2	01/08/16		C-TEST1	11	1	21	Droite
3	TESTOTOLITHE2	1	2	01/08/16		C-TEST2	11	1	21	Gauche
4	TESTOTOLITHE3	1	2	01/08/16		C-TEST3	11	1	22	central
5	TESTOTOLITHE4	1	2	01/08/16		C-TEST4	11	1	23	central droit
6						TRANSECT-08/16-1	8	3		
7						TRANSECT-08/16-2	8	3		
8						TRANSECT-08/16-3	8	3		
9						TRANSECT-08/16-4	8	3		
10	TESTALOSON1	3	1	01/09/16						
11	TESTOTOLITHE3	2	2	01/08/16		C-TEST3	25	1	22	central
12	TESTOTOLITHE4	1	4	01/08/16		C-TEST4	11	6	4000	central droit
13		1	2				11	6		

FIGURE 4.3 – Exemple d'un fichier CSV

Dans cet exemple, l'import ne sera pas réalisé pour les raisons suivantes :

- en ligne 12, le numéro de container n'existe pas ;
- la ligne 13 ne contient ni numéro d'échantillon, ni de numéro de container.

Sans tenir compte des erreurs, voici les opérations qui seraient exécutées :

- lignes 2 à 5, 11 et 12 : création d'échantillons, avec création du mouvement d'entrée dans les containers correspondants ;
- lignes 6 à 9 : création de containers ;
- ligne 10 : création d'un échantillon non rangé ;

4.2 Export de données au format JSON

Il est possible d'exporter un lot de données, réparties sur plusieurs tables, pour les réimporter dans une autre base de données. Par défaut, l'export « technique » des modèles d'exports est disponible.

Les données sont exportées au format JSON.

4.2.1 Description des modèles d'export

Choisissez le menu *Paramètres > Modèles d'export*. Hormis le nom du modèle, toutes les autres informations permettent de décrire les tables à exporter et les relations entre elles.

La saisie d'une nouvelle table passe par l'ajout d'un nouvel item dans la partie *Description du modèle*. Chaque item peut être déplacé après création, si nécessaire.

Pour chaque table à exporter, voici les informations à renseigner :

- nom de la table, telle qu'il figure dans la base de données
- alias de la table : si une même table peut être reliée à des tables parentes différentes, cette colonne devra être renseignée avec un nom différent pour chaque instance.
- clé primaire : indiquez la clé primaire utilisée dans la table. Elle ne doit pas être renseignée dans le cas d'une table porteuse d'une relation n-n, dont la clé est composée des clés des deux tables parentes.

- clé métier : il s'agit de la colonne qui permet de retrouver de manière unique un enregistrement dans la table. Selon les cas de figure, il peut s'agir :
 - du libellé, pour une table de paramètres
 - de la clé primaire elle-même, pour certaines tables de paramètres dont la clé est significative. Cela permet de conserver la valeur de cette clé, même si le libellé change
 - du champ UUID, qui est un identifiant technique généré avec un algorithme garantissant qu'il est unique au niveau mondial. Cette valeur sera utilisée chaque fois qu'elle est disponible
- clé étrangère : le nom du champ porteur de la relation avec le parent, qui contient donc la clé du parent
- la liste des champs de type booléen, en raison d'une particularité lors des importations liée à ceux-ci
- la liste des alias (ou du nom des tables) des tables filles
- dans le cas d'une table porteuse d'une relation n-n, c'est à dire dont la clé est composée des clés des deux tables parentes, il faudra indiquer :
 - le nom du champ comprenant la seconde clé étrangère
 - l'alias (ou le nom de la table) de la seconde table parente

La première table présente dans la liste doit être la table principale de l'export. Les tables filles doivent être créées après leurs tables parentes.

Il est possible d'indiquer plusieurs tables principales (sans parents) dans le même modèle.

4.2.2 Importer un fichier JSON

Il est possible de réaliser une importation rapide depuis le menu de création des modèles d'exportation, depuis le détail d'un modèle. Cette fonction peut être pratique pour mettre à jour des tables de référence.

Annexe A

Mettre en place une réplication de la base postgresql vers un autre serveur

A.1 Présentation

L'objectif de ce chapitre est de présenter comment mettre en œuvre une réplication entre deux serveurs Postgresql, pour éviter toute perte accidentelle d'un enregistrement.

Il a été écrit par Alexandra Darrieutort, stagiaire à Irstea en 2016, et complété par Jacques Foury, responsable informatique du centre Irstea de Cestas (33), qui se sont inspirés de divers documents [18] [19] [20] [21].

A.1.1 Besoins exprimés

Mise en place d'une réplication d'un serveur postgresSQL de sorte qu'il y ait préservation des données, c'est-à-dire qu'une écriture faite sur le serveur maître se retrouve sur le serveur esclave. Le besoin en haute disponibilité n'est pas primordial.

A.1.2 Principe

Le mode de réplication correspondant au besoin est *maître/esclave*. On peut lire et écrire sur le maître et seulement lire sur l'esclave s'il est configuré en *hot standby*. Ici, le serveur maître est *citerne-8* et le serveur esclave est *chappie*.

Les modifications de données sont enregistrées dans des journaux de transactions appelés **WAL (Write-Ahead Log) xlogs**. Ces WAL sont transférés à l'esclave qui les rejoue continuellement de sorte à se retrouver dans le même état que le maître. Il sera alors prêt à prendre la relève en cas d'indisponibilité du maître.

Grâce au principe de *Streaming replication*, on n'attend plus que le fichier WAL (16 Mio) soit rempli mais il sera transmis sans délai du maître à l'esclave.

A.1.3 Limitations et précautions

Dans la configuration, comme on va conserver 256 xlogs à l'aide du paramètre **wal_keep_segments**, il faut prévoir assez d'espace disque disponible.

La réplication entre deux serveurs de versions différentes de postgresql est impossible.

A.2 Mise à jour du serveur (version 9.3) en version 9.4 dans *citerne-8* :

On installe la dernière version de postgresql, on liste les clusters qui tournent et on supprime le cluster 9.4 existant :

```
root@citerne-8:~# apt-get install postgresql-9.4
root@citerne-8:~# pg_lsclusters
root@citerne-8:~# pg_dropcluster --stop 9.4 main
```

Mise à jour du cluster :

```
root@citerne-8:~# pg_upgradecluster 9.3 main
```

Liste des clusters et visualisation de leur activité :

```
root@citerne-8:~# pg_lsclusters
```

Suppression de l'ancien cluster :

```
root@citerne-8:~# pg_dropcluster --stop 9.3 main
```

Modification du port du cluster 9.4 dans le fichier */etc/postgresql/9.4/main/postgresql.conf* :

```
port = 5432
```

A.3 Installation de PostgreSQL sur *chappie* et mise en place des clés ssh

```
root@chappie:~# apt-get install postgresql-9.4
root@chappie:~# su - postgres
postgres@chappie:~$ mkdir /var/lib/postgresql/.ssh/
postgres@chappie:~$ ssh-keygen
```

Pour la connexion ssh entre les deux serveurs, il faut mettre la clé de l'utilisateur postgres contenue dans le fichier **id_rsa.pub** sur *chappie* dans le fichier **authorized_keys** de *citerne-8* et inversement.

A.4 Mise en place de la réplication

A.4.1 Maître

Création de l'utilisateur postgresql chargé de la réplication :

```
root@citerne-8:~# su - postgres
postgres:~$ psql -c "CREATE USER rep REPLICATION LOGIN ENCRYPTED
PASSWORD 'desperados';"
```

Dans le fichier **pg_hba.conf** (*/etc/postgresql/9.4/main/*) ajoutez :

```
host      replication      rep      10.33.192.31/32      md5
```

Pour le paramètre **wal_keep_segments**, on lui donne une valeur assez grande pour éviter d'accumuler un retard trop important entre les deux serveurs en cas d'indisponibilité de l'esclave.

Dans le fichier **postgresql.conf** ajoutez ces lignes¹ :

1. Attention : Si vous faites un copier-coller, les apostrophes ne sont pas des apostrophes droites donc il faudra les modifier

ANNEXE A. METTRE EN PLACE UNE RÉPLICATION DE LA BASE POSTGRESQL VERS UN AUTRE SERVEUR

```
listen_address = 'localhost,10.33.192.36'
wal_level = hot_standby
max_wal_senders = 3
max_wal_size = 436MB
wal_keep_segments = 256
```

Redémarrez ensuite le service postgresql.

A.4.2 Esclave

Arrêtez le service postgresql, puis ajoutez ces lignes dans le fichier **postgresql.conf** :

```
wal_level = hot_standby
max_wal_senders = 3
max_wal_size = 384MB
wal_keep_segments = 256
hot_standby = on
max_locks_per_transaction = 128
```

Modifiez le fichier **pg_hba.conf** :

```
host      replication      rep      10.33.192.36/32 md5
```

Effectuez la sauvegarde complète des bases du serveur maître (depuis l'esclave, toujours) avec l'utilisateur postgres :

```
pg_dropcluster 9.5 main
pg_basebackup -h 10.33.192.36 -D /var/lib/postgresql/9.5/main -U rep -v
-P --xlog
```

L'option `--xlog` est ajoutée pour garder les derniers journaux de transactions.

Créez le fichier **recovery.conf** dans `/var/lib/postgresql/9.5/main/` pour configurer la restauration continue.

La restauration en continu s'active à l'aide du paramètre *standby_mode*. Pour se connecter au maître et récupérer les WAL, on définit les informations nécessaires dans le paramètre *primary_conninfo*.

Le paramètre *trigger_file* indique si la restauration doit être interrompue (si le fichier indiqué est présent, le processus est arrêté).

```
standby_mode = on
primary_conninfo = 'host=10.33.192.36 port=5432 user=rep password=
desperados'
trigger_file = '/var/lib/postgresql/9.4/postgresql.trigger'
```

Pour finir, démarrez le service postgresql.

A.5 Informations de monitoring

Le fichier de logs **postgresql-9.4-main.log** se trouve dans le répertoire `/var/log/postgresql/`

Pour savoir où en est la réplication du côté du maître :

```
sudo -u postgres psql -x -c "select * from pg_stat_replication;"
```

Pour savoir à quand remonte la dernière synchronisation du côté de l'esclave :

```
sudo -u postgres psql -x -c "SELECT now() -
pg_last_xact_replay_timestamp() AS time_lag;"
```

Pour voir le numéro du snapshot actuel :

```
sudo -u postgres psql -x -c "SELECT txid_current_snapshot();" 
```

A.6 Pour tester le failover ou gérer un interruption

Le serveur maître est indisponible.

Il faut arrêter la restauration continue sur l'esclave pour qu'il devienne le maître, en créant le fichier *trigger*. Les bases vont alors passer en mode read/write et le fichier *recovery.conf* sera renommé *recovery.done*.

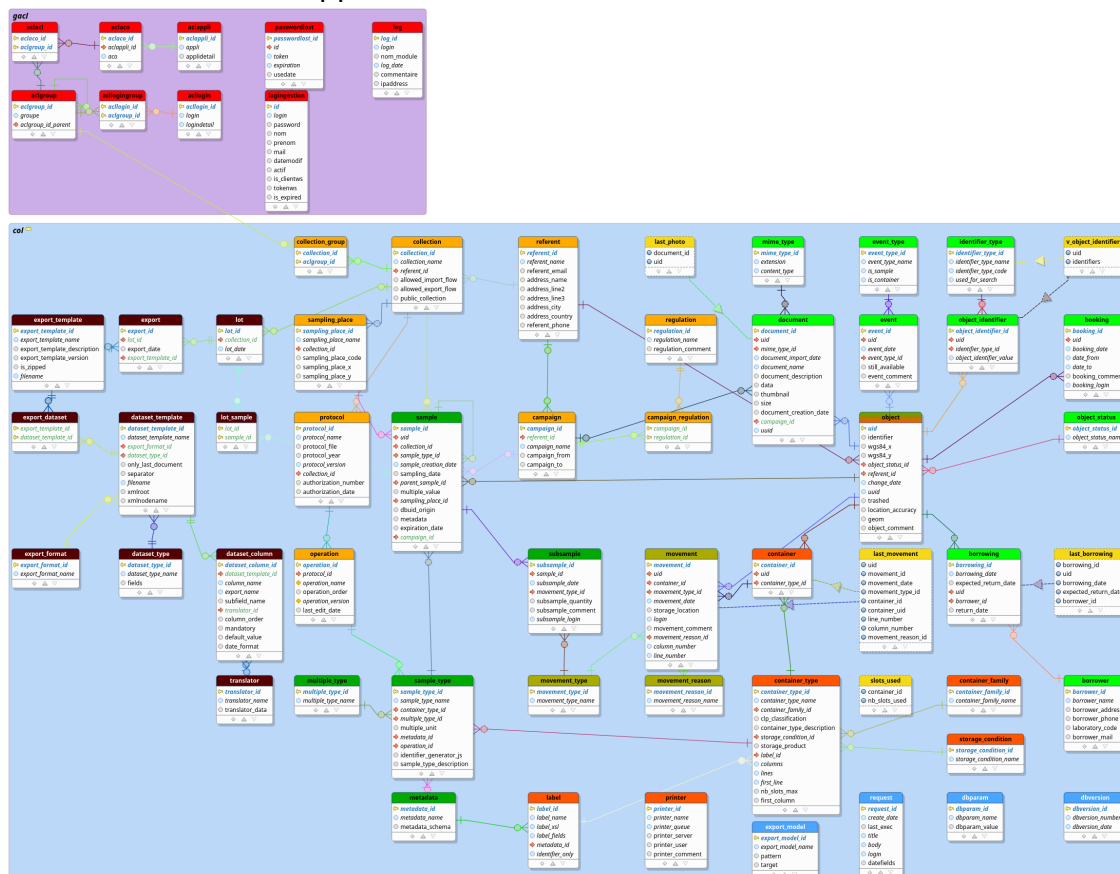
```
sudo touch /var/lib/postgresql/9.4/postgresql.trigger
```

Lorsque le maître sera de retour, la réplication ne fonctionnera plus. Vous devrez restaurer les données provenant du serveur esclave dans le serveur maître, puis relancer la réplication, en recréant le fichier *recovery.conf*, comme décrit dans la section A.4.2 *Esclave*.

Annexe B

Structure de la base de données

La structure et le détail des tables peut être consulté directement depuis le menu *Paramètres* de l'application.



B.1 Description des tables

B.1.1 Schema col

Tables

booking Table of object's bookings

Column name	Type	Not null	Key	Comment
booking_id	integer	X	X	
uid	integer	X		
booking_date	timestamp without time zone	X		Date of booking
date_from	timestamp without time zone	X		Date-time of booking start
date_to	timestamp without time zone	X		Date-time of booking end
booking_comment	character varying			Comment
booking_login	character varying	X		Login used to perform the reservation

References uid : col.object (uid)

borrower List of borrowers

Column name	Type	Not null	Key	Comment
borrower_id	integer	X	X	
borrower_name	character varying	X		
borrower_address	character varying			Address of the borrower
borrower_phone	character varying			Phone of the contact of the borrower
laboratory_code	character varying			Laboratory code of the borrower
borrower_mail	character varying			Mail of the borrower

Referenced by borrower_id : col.borrowing (borrower_id)

borrowing List of borrowings

ANNEXE B. STRUCTURE DE LA BASE DE DONNÉES

Column name	Type	Not null	Key	Comment
borrowing_id	integer	X	X	Date of the borrowing
borrowing_date	timestamp without time zone	X		
expected_return_date	timestamp without time zone			
uid	integer	X		Expected return date of the object
borrower_id	integer	X		
return_date	timestamp without time zone			

References borrowing_id : col.borrower (borrower_id)
 uid : col.object (uid)

campaign List of sampling campaigns

Column name	Type	Not null	Key	Comment
campaign_id	integer	X	X	Name of the campaign
referent_id	integer			
campaign_name	character varying	X		
campaign_from	timestamp without time zone			Date of start of the campaign
campaign_to	timestamp without time zone			date of end of the campaign

References referent_id : col.referent (referent_id)

Referenced by campaign_id : col.campaign_regulation (campaign_id)
 campaign_id : col.document (campaign_id)
 campaign_id : col.sample (campaign_id)

campaign_regulation List of regulations attached to a campaign

Column name	Type	Not null	Key	Comment
campaign_id	integer	X	X	
regulation_id	integer	X	X	

References campaign_id : col.campaign (campaign_id)
 regulation_id : col.regulation (regulation_id)

collection List of all collections into the database

Column name	Type	Not null	Key	Comment
collection_id	integer	X	X	Allow an external source to update a collection Allow interrogation requests from external sources Set if a collection can be requested without authentication List of keywords, separated by a comma Name used to communicate from the collection, in export module by example
collection_name	character varying	X		
referent_id	integer			
allowed_import_flow	boolean			
allowed_export_flow	boolean			
public_collection	boolean			
collection_keywords	character varying			
collection_displayname	character varying			
license_id	integer			

References license_id : col.license (license_id)
 referent_id : col.referent (referent_id)

Referenced by collection_id : col.lot (collection_id)
 collection_id : col.samplesearch (collection_id)
 collection_id : col.sampling_place (collection_id)
 collection_id : col.collection_group (collection_id)
 collection_id : col.protocol (collection_id)
 collection_id : col.sample (collection_id)

collection_group Table of project approvals

Column name	Type	Not null	Key	Comment
collection_id	integer	X	X	
aclgroup_id	integer	X	X	

References collection_id : col.collection (collection_id)
 aclgroup_id : gacl.aclgroup (aclgroup_id)

container Liste of containers

Column name	Type	Not null	Key	Comment
container_id	integer	X	X	
uid	integer	X		
container_type_id	integer	X		

References container_type_id : col.container_type (container_type_id)
 uid : col.object (uid)

Referenced by container_id : col.movement (container_id)

container_family General family of containers

Column name	Type	Not null	Key	Comment
container_family_id	integer	X	X	
container_family_name	character varying	X		

Referenced by container_family_id : col.container_type (container_family_id)

container_type Table of types of containers

Column name	Type	Not null	Key	Comment
container_type_id	integer	X	X	
container_type_name	character varying	X		
container_family_id	integer	X		
clp_classification	character varying			Risk classification according to the European CLP directive
container_type_description	character varying			Long description
storage_condition_id	integer			
storage_product	character varying			Product used for storage (formol, alcohol...)
label_id	integer			
columns	integer	X		Number of storage columns in the container
lines	integer	X		Number of storage lines in the container
first_line	character varying	X		Place of the first line : T : top B : bottom
nb_slots_max	integer			Number maximum of slots in the container
first_column	character varying			Place of the first column : L : left R : Right

References container_family_id : col.container_family (container_family_id)

label_id : col.label (label_id)

storage_condition_id : col.storage_condition (storage_condition_id)

Referenced by container_type_id : col.container (container_type_id)

container_type_id : col.sample_type (container_type_id)

country List of the countries

Column name	Type	Not null	Key	Comment
country_id	integer	X	X	Numeric ISO code of the country
country_name	character varying	X		Name of the country
country_code2	character varying(2)	X		Code of the country, on 2 positions
country_code3	character varying(3)			Code of the country, on 3 positions

Referenced by country_id : col.sample (country_id)
country_id : col.sampling_place (country_id)

dataset_column List of columns included into the dataset

Column name	Type	Not null	Key	Comment
dataset_column_id	integer	X	X	
dataset_template_id	integer	X		
column_name	character varying	X		Name of the column into the database
export_name	character varying	X		Name of the column into the export file
subfield_name	character varying			Name of the field if it into the metadata description of the sample or secondary identifier, etc.
translator_id	integer			
column_order	smallint			order of displaying in the exported file
mandatory	boolean			Is the content of the column is mandatory to export data ?
default_value	character varying			Default value, if the value is not filled in
date_format	character varying			Export date format, in php notation. Example : d/m/Y H :i :s for 25/12/2020 17 :15 :00

References dataset_template_id : col.dataset_template (dataset_template_id)
translator_id : col.translator (translator_id)

dataset_template List of templates of dataset

ANNEXE B. STRUCTURE DE LA BASE DE DONNÉES

Column name	Type	Not null	Key	Comment
dataset_template_id	integer	X	X	Name of the template
dataset_template_name	character varying	X		
export_format_id	integer	X		
dataset_type_id	integer	X		
only_last_document	boolean			
separator	character varying	X		If type document, specify if only the last document attached to a sample is exported
filename	character varying			Separator used in csv files (tab ,;)
xmlroot	character varying			File name generated, with extension
xmlnodename	character varying			xml root to generate
xslcontent	character varying			Name of a node in a xml file, for list of samples by example
				Transformation of the generated xml to create a specific xml file

References dataset_type_id : col.dataset_type (dataset_type_id)
 export_format_id : col.export_format (export_format_id)

Referenced by dataset_template_id : col.dataset_column (dataset_template_id)
 dataset_template_id : col.export_dataset (dataset_template_id)

dataset_type Origine of the dataset : sample, collection, document

Column name	Type	Not null	Key	Comment
dataset_type_id	integer	X	X	List of allowed fields of the database (json array)
dataset_type_name	character varying	X		
fields	json			

Referenced by dataset_type_id : col.dataset_template (dataset_type_id)

dbparam Table of parameters intrinsically associated to the instance

Column name	Type	Not null	Key	Comment
dbparam_id	integer	X	X	Name of the parameter
dbparam_name	character varying	X		
dbparam_value	character varying			Value of the parameter

dbversion Table of the database versions

Column name	Type	Not null	Key	Comment
dbversion_id	integer	X	X	Number of the version
dbversion_number	character varying	X		
dbversion_date	timestamp without time zone	X		Date of the version

document Numeric docs associated to an objet or a campaign

Column name	Type	Not null	Key	Comment
document_id	integer	X	X	Import date into the data-base
uid	integer			
mime_type_id	integer	X		
document_import_date	timestamp without time zone	X		
document_name	character varying	X		Original name
document_description	character varying			Description
data	bytea			Binary content (object imported)
thumbnail	bytea			Thumbnail in PNG format (only for pdf, jpg or png docs)
size	integer			Size of downloaded file
document_creation_date	timestamp without time zone			Create date of the document (date of photo shooting, for example)
campaign_id	integer			
uuid	uuid	X		

References campaign_id : col.campaign (campaign_id)

mime_type_id : col.mime_type (mime_type_id)

uid : col.object (uid)

event Table of events

Column name	Type	Not null	Key	Comment
event_id	integer	X	X	Date-time of the event
uid	integer	X		
event_date	timestamp without time zone	X		
event_type_id	integer	X		still available content in the object, after the event Comment
still_available	character varying			
event_comment	character varying			

References event_type_id : col.event_type (event_type_id)

uid : col.object (uid)

event_type Event types table

Column name	Type	Not null	Key	Comment
event_type_id	integer	X	X	Name of the type of event
event_type_name	character varying	X		
is_sample	boolean	X		The event is applicable to the samples
is_container	boolean	X		The event is applicable to the containers

Referenced by event_type_id : col.event (event_type_id)

export List of exports processed

Column name	Type	Not null	Key	Comment
export_id	integer	X	X	Date of last execution of the export
lot_id	integer	X		
export_date	timestamp without time zone			
export_template_id	integer	X		

References export_template_id : col.export_template (export_template_id)
lot_id : col.lot (lot_id)

export_dataset List of datasets embedded into the template of export

Column name	Type	Not null	Key	Comment
export_template_id	integer	X	X	
dataset_template_id	integer	X	X	

References dataset_template_id : col.dataset_template (dataset_template_id)
export_template_id : col.export_template (export_template_id)

export_format List of formats of export

Column name	Type	Not null	Key	Comment
export_format_id	integer	X	X	
export_format_name	character varying	X		

Referenced by export_format_id : col.dataset_template (export_format_id)

export_model Structure of an export/import of table data

Column name	Type	Not null	Key	Comment
export_model_id	integer	X	X	Name of the structure of export Pattern of the export/import. Structure : [technical-Key :string,businessKey :string,tableName :string] Main table targetted
export_model_name	character varying	X		
pattern	json			
target	character varying			

export_template List of models of export

Column name	Type	Not null	Key	Comment
export_template_id	integer	X	X	Name of the model
export_template_name	character varying	X		
export_template_description	character varying			Description of the model
export_template_version	character varying			Version of the model, if necessary
is_zipped	boolean			Specify if the generated files must be zipped
filename	character varying	X		Name of the file generated

Referenced by export_template_id : col.export (export_template_id)
 export_template_id : col.export_dataset (export_template_id)

identifier_type Table of identifier types

Column name	Type	Not null	Key	Comment
identifier_type_id	integer	X	X	Textual name of the identifier
identifier_type_name	character varying	X		
identifier_type_code	character varying	X		Identifier code, used in the labels
used_for_search	boolean	X		Is the identifier usable for barcode searches ?

Referenced by identifier_type_id : col.object_identifier (identifier_type_id)

label Table of label models

ANNEXE B. STRUCTURE DE LA BASE DE DONNÉES

Column name	Type	Not null	Key	Comment
label_id	integer	X	X	
label_name	character varying	X		Name of the model
label_xsl	character varying	X		XSL content used by FOP transformation (https://xmlgraphics.apache.org/fop/)
label_fields	character varying	X		List of fields incorporated in the QR CODE
metadata_id	integer			Model of the metadata template associated with this label
identifier_only	boolean	X		true : the qrcode contains only a business identifier

References metadata_id : col.metadata (metadata_id)

Referenced by label_id : col.container_type (label_id)

license List of licenses usable to communicate informations on the collections

Column name	Type	Not null	Key	Comment
license_id	integer	X	X	
license_name	character varying	X		Name of the license
license_url	character varying			Link of download of the text of the license

Referenced by license_id : col.collection (license_id)

lot List of lots of export

Column name	Type	Not null	Key	Comment
lot_id	integer	X	X	
collection_id	integer			
lot_date	timestamp without time zone	X		Date of creation of the lot

References collection_id : col.collection (collection_id)

Referenced by lot_id : col.export (lot_id)

lot_id : col.lot_sample (lot_id)

lot_sample List of samples associated into a lot

Column name	Type	Not null	Key	Comment
lot_id	integer	X	X	
sample_id	integer	X	X	

References lot_id : col.lot (lot_id)
sample_id : col.sample (sample_id)

metadata Table of metadata usable with types of samples

Column name	Type	Not null	Key	Comment
metadata_id	integer	X	X	
metadata_name	character varying	X		Name of the metadata set
metadata_schema	json			JSON schema of the meta- data form

Referenced by metadata_id : col.label (metadata_id)
metadata_id : col.sample_type (metadata_id)

mime_type Mime types of imported files

Column name	Type	Not null	Key	Comment
mime_type_id	integer	X	X	
extension	character varying	X		File extension
content_type	character varying	X		Official mime type

Referenced by mime_type_id : col.document (mime_type_id)

movement Records of objects movements

Column name	Type	Not null	Key	Comment
movement_id	integer	X	X	
uid	integer	X		
container_id	integer			
movement_type_id	integer	X		
movement_date	timestamp without time zone	X		Date-time of the movement
storage_location	character varying			Place of the object in the container, in textual form
login	character varying	X		Name of the operator who performed the operation
movement_comment	character varying			Comment
movement_reason_id	integer			
column_number	integer	X		Number of the storage co- lumn in the container
line_number	integer	X		Number of the storage line in the container

References container_id : col.container (container_id)
movement_reason_id : col.movement_reason (movement_reason_id)
movement_type_id : col.movement_type (movement_type_id)
uid : col.object (uid)

movement_reason List of the reasons of the movement

Column name	Type	Not null	Key	Comment
movement_reason_id	integer	X	X	
movement_reason_name	character varying	X		

Referenced by movement_reason_id : col.movement (movement_reason_id)

movement_type Type de mouvement

Column name	Type	Not null	Key	Comment
movement_type_id	integer	X	X	
movement_type_name	character varying	X		

Referenced by movement_type_id : col.movement (movement_type_id)

movement_type_id : col.subsample (movement_type_id)

multiple_type Table of categories of potential sub-sampling (unit, quantity, percentage, etc.)

Column name	Type	Not null	Key	Comment
multiple_type_id	integer	X	X	
multiple_type_name	character varying	X		

Referenced by multiple_type_id : col.sample_type (multiple_type_id)

object Table of objects

Column name	Type	Not null	Key	Comment
uid	integer	X	X	Unique identifier in the database of all objects
identifier	character varying			Main working identifier
wgs84_x	double precision			GPS longitude, in decimal form
wgs84_y	double precision			GPS latitude, in decimal form
object_status_id	integer			
referent_id	integer			
change_date	timestamp without time zone	X		Technical date of change of the object
uuid	uuid	X		UUID of the object
trashed	boolean			If the object is trashed before completely destroyed?
location_accuracy	double precision			Location accuracy of the object, in meters
geom	geography(Point,4326)			Geographic point generate from wgs84_x and wgs84_y
object_comment	character varying			Comment on the object (sample or container)

References object_status_id : col.object_status (object_status_id)
referent_id : col.referent (referent_id)

Referenced by uid : col.booking (uid)
uid : col.container (uid)
uid : col.document (uid)
uid : col.event (uid)
uid : col.borrowing (uid)
uid : col.object_identifier (uid)
uid : col.sample (uid)
uid : col.movement (uid)

object_identifier Table of complementary identifiers

Column name	Type	Not null	Key	Comment
object_identifier_id	integer	X	X	
uid	integer	X		
identifier_type_id	integer	X		
object_identifier_value	character varying	X		Identifier value

References identifier_type_id : col.identifier_type (identifier_type_id)
uid : col.object (uid)

object_status Table of types of status

Column name	Type	Not null	Key	Comment
object_status_id	integer	X	X	
object_status_name	character varying	X		

Referenced by object_status_id : col.object (object_status_id)

operation List of operations attached to a protocol

Column name	Type	Not null	Key	Comment
operation_id	integer	X	X	
protocol_id	integer	X		
operation_name	character varying	X		
operation_order	integer			Order to perform the operation in the protocol
operation_version	character varying			Version of the operation
last_edit_date	timestamp without time zone			Last edit date of the operation

References protocol_id : col.protocol (protocol_id)

Referenced by operation_id : col.sample_type (operation_id)

printer Table of printers directly managed by the server

Column name	Type	Not null	Key	Comment
printer_id	integer	X	X	
printer_name	character varying	X		Usual name of the printer, displayed in the forms
printer_queue	character varying	X		Name of the printer known by the operating system of the server
printer_server	character varying			Server address, if the printer is not connected at the localhost
printer_user	character varying			User used to print, if necessary
printer_comment	character varying			Comment

protocol List of protocols

Column name	Type	Not null	Key	Comment
protocol_id	integer	X	X	
protocol_name	character varying	X		Name of the protocol
protocol_file	bytea			PDF description of the protocol
protocol_year	smallint			Year of the protocol
protocol_version	character varying	X		Version of the protocol
collection_id	integer			
authorization_number	character varying			Number of the prelevement authorization
authorization_date	timestamp without time zone			Date of the prelevement authorization

References collection_id : col.collection (collection_id)

Referenced by protocol_id : col.operation (protocol_id)

referent Table of sample referents

Column name	Type	Not null	Key	Comment
referent_id	integer	X	X	
referent_name	character varying	X		Name, firstname-lastname or department name
referent_email	character varying			Email for contact
address_name	character varying			Name for postal address
address_line2	character varying			second line in postal address
address_line3	character varying			third line in postal address
address_city	character varying			ZIPCode and City in postal address
address_country	character varying			Country in postal address
referent_phone	character varying			Contact phone
referent_firstname	character varying			Firstname of the referent
academical_directory	character varying			Academical directory used to identifier the referent, as https://orcid.org or https://www.researchgate.net
academical_link	character varying			Link used to identify the referent, as https://orcid.org/0000-0003-4207-4107

Referenced by referent_id : col.collection (referent_id)

referent_id : col.campaign (referent_id)

referent_id : col.object (referent_id)

regulation Table of regulations

Column name	Type	Not null	Key	Comment
regulation_id	integer	X	X	Name of the regulation regulatory clarity
regulation_name	character varying	X		
regulation_comment	character varying			

Referenced by regulation_id : col.campaign_regulation (regulation_id)

request Request table in database

Column name	Type	Not null	Key	Comment
request_id	integer	X	X	Date of create of the request
create_date	timestamp without time zone	X		
last_exec	timestamp without time zone			Date of the last execution
title	character varying	X		Title of the request
body	character varying	X		Body of the request. Don't begin it by SELECT, which will be added automatically
login	character varying	X		Login of the creator of the request
datefields	character varying			List of the date fields used in the request, separated by a comma, for format it

sample Table of samples

Column name	Type	Not null	Key	Comment
sample_id	integer	X	X	
uid	integer	X		
collection_id	integer	X		
sample_type_id	integer	X		
sample_creation_date	timestamp without time zone	X		Creation date of the record in the database
sampling_date	timestamp without time zone			Creation date of the phy- sical sample or date of sampling
parent_sample_id	integer			
multiple_value	double precision			
sampling_place_id	integer			
dbuid_origin	character varying			Reference used in the ori- ginal database, under the form db :uid. Used for read the labels created in others instances
metadata	json			Metadata associated with the sample, in JSON for- mat
expiration_date	timestamp without time zone			Date of expiration of the sample. After this date, the sample is not usable
campaign_id	integer			
country_id	integer			

References campaign_id : col.campaign (campaign_id)
collection_id : col.collection (collection_id)
country_id : col.country (country_id)
uid : col.object (uid)
parent_sample_id : col.sample (sample_id)
sample_type_id : col.sample_type (sample_type_id)
sampling_place_id : col.sampling_place (sampling_place_id)

Referenced by sample_id : col.lot_sample (sample_id)
sample_id : col.sample (parent_sample_id)
sample_id : col.subsample (sample_id)

sample_type Table of the types of samples

ANNEXE B. STRUCTURE DE LA BASE DE DONNÉES

Column name	Type	Not null	Key	Comment
sample_type_id	integer	X	X	Name of the type
sample_type_name	character varying	X		
container_type_id	integer			Name of the unit used to qualify the number of sub-samples (ml, number, g, etc.)
multiple_type_id	integer			
multiple_unit	character varying			
metadata_id	integer			Javascript function code used to automatically generate a working identifier from the intered information
operation_id	integer			
identifier_generator_js	character varying			
sample_type_description	character varying			Description of the type of sample

References container_type_id : col.container_type (container_type_id)
 metadata_id : col.metadata (metadata_id)
 multiple_type_id : col.multiple_type (multiple_type_id)
 operation_id : col.operation (operation_id)

Referenced by sample_type_id : col.sample (sample_type_id)

samplesearch List of sample saved search

Column name	Type	Not null	Key	Comment
samplesearch_id	integer	X		Name of the search parameters
samplesearch_name	character varying	X		
samplesearch_data	json			List of all research parameters
samplesearch_login	character varying			Login of the creator
collection_id	integer			

References collection_id : col.collection (collection_id)

sampling_place Table of sampling places

Column name	Type	Not null	Key	Comment
sampling_place_id	integer	X	X	Name of the sampling place Collection of rattachment Working code of the station
sampling_place_name	character varying	X		
collection_id	integer			
sampling_place_code	character varying			Longitude of the station, in WGS84 Latitude of the station, in WGS84
sampling_place_x	double precision			
sampling_place_y	double precision			
country_id	integer			

References collection_id : col.collection (collection_id)
country_id : col.country (country_id)

Referenced by sampling_place_id : col.sample (sampling_place_id)

storage_condition Table of the conditions of storage

Column name	Type	Not null	Key	Comment
storage_condition_id	integer	X	X	
storage_condition_name	character varying	X		

Referenced by storage_condition_id : col.container_type (storage_condition_id)

subsample Table of sub-sample takes and returns

Column name	Type	Not null	Key	Comment
subsample_id	integer	X	X	Date-time of the operation
sample_id	integer	X		
subsample_date	timestamp without time zone	X		
movement_type_id	integer	X		Quantity taken or returned
subsample_quantity	double precision			
subsample_comment	character varying			Comment on this operation
subsample_login	character varying	X		Login of the user who per- form this operation

References movement_type_id : col.movement_type (movement_type_id)
sample_id : col.sample (sample_id)

translator Table of translator of values

ANNEXE B. STRUCTURE DE LA BASE DE DONNÉES

Column name	Type	Not null	Key	Comment
translator_id	integer	X	X	List of translations under the form term_in_database :term_in_the_export_file
translator_name	character varying	X		
translator_data	json			

Referenced by translator_id : col.dataset_column (translator_id)

Views

last_borrowing SELECT b1.borrowing_id, b1.uid, b1.borrowing_date, b1.expected_return_date, b1.borrower_id FROM col.borrowing b1 WHERE (b1.borrowing_id = (SELECT b2.borrowing_id FROM col.borrowing b2 WHERE ((b1.uid = b2.uid) AND (b2.return_date IS NULL)) ORDER BY b2.borrowing_date DESC LIMIT 1));

Column name	Type
borrowing_id	integer
uid	integer
borrowing_date	timestamp without time zone
expected_return_date	timestamp without time zone
borrower_id	integer

last_movement SELECT s.uid, s.movement_id, s.movement_date, s.movement_type_id, s.container_id, c.uid AS container_uid, s.line_number, s.column_number, s.movement_reason_id FROM (col.movement s LEFT JOIN col.container c USING (container_id)) WHERE (s.movement_id = (SELECT st.movement_id FROM col.movement st WHERE (s.uid = st.uid) ORDER BY st.movement_date DESC LIMIT 1));

Column name	Type
uid	integer
movement_id	integer
movement_date	timestamp without time zone
movement_type_id	integer
container_id	integer
container_uid	integer
line_number	integer
column_number	integer
movement_reason_id	integer

last_photo SELECT d.document_id, d.uid FROM col.document d WHERE (d.document_id = (SELECT d1.document_id FROM col.document d1 WHERE ((d1.mime_type_id = ANY (ARRAY[4, 5, 6])) AND (d.uid = d1.uid)) ORDER BY d1.document_creation_date DESC, d1.document_import_date DESC, d1.document_id DESC LIMIT 1));

Column name	Type
document_id	integer
uid	integer

slots_used SELECT last_movement.container_id, count(*) AS nb_slots_used
FROM col.last_movement WHERE (last_movement.movement_type_id = 1) GROUP
BY last_movement.container_id;

Column name	Type
container_id	integer
nb_slots_used	bigint

v_object_identifier SELECT object_identifier.uid, array_to_string(array_agg((((identifier_type.id
|| ' ' : :text) || (object_identifier.object_identifier_value) : :text) ORDER BY identi-
fier_type.identifier_type_code, object_identifier.object_identifier_value), ' ' : :text)
AS identifiers FROM (col.object_identifier JOIN col.identifier_type USING (identi-
fier_type_id)) GROUP BY object_identifier.uid ORDER BY object_identifier.uid;

Column name	Type
uid	integer
identifiers	text

B.1.2 Schema gacl

Tables

aclacl Table of rights granted

Column name	Type	Not null	Key	Comment
aclaco_id	integer	X	X	
aclgroup_id	integer	X	X	

References aclaco_id : gacl.aclaco (aclaco_id)
aclgroup_id : gacl.aclgroup (aclgroup_id)

aclaco Table of managed rights

Column name	Type	Not null	Key	Comment
aclaco_id	integer	X	X	
aclappli_id	integer	X		
aco	character varying	X		Name of the right in the ap- plication

References aclappli_id : gacl.aclappli (aclappli_id)

Referenced by aclaco_id : gacl.aclacl (aclaco_id)

aclappli Table of managed applications

Column name	Type	Not null	Key	Comment
aclappli_id	integer	X	X	
appli	character varying	X		Code of the application used to manage the rights
applidetail	character varying			Description of the applica- tion

Referenced by aclappli_id : gacl.aclaco (aclappli_id)

aclgroup Groups of logins

Column name	Type	Not null	Key	Comment
aclgroup_id	integer	X	X	Name of the group
groupe	character varying	X		
aclgroup_id_parent	integer			Parent group who inherits of the rights attributed to this group

References aclgroup_id_parent : gacl.aclgroup (aclgroup_id)

Referenced by aclgroup_id : gacl.aclacl (aclgroup_id)

aclgroup_id : gacl.aclgroup (aclgroup_id_parent)

aclgroup_id : gacl.acloggingroup (aclgroup_id)

aclgroup_id : col.collection_group (aclgroup_id)

aclogin List of logins granted to access to the modules of the application

Column name	Type	Not null	Key	Comment
aclogin_id	integer	X	X	Login. It must be the same as the table loggingestion Displayed name
login	character varying	X		
logindetail	character varying	X		

Referenced by aclogin_id : gacl.acloggingroup (aclogin_id)

acloggingroup List of logins in the groups

Column name	Type	Not null	Key	Comment
aclogin_id	integer	X	X	
aclgroup_id	integer	X	X	

References aclgroup_id : gacl.aclgroup (aclgroup_id)

aclogin_id : gacl.aclogin (aclogin_id)

log List of all recorded operations (logins, calls of modules, etc.)

Column name	Type	Not null	Key	Comment
log_id	integer	X	X	
login	character varying(32)	X		Code of the login who performs the operation
nom_module	character varying			Name of the performed module
log_date	timestamp without time zone	X		Date-time of the operation
commentaire	character varying			Complementary data recorded
ipaddress	character varying			IP address of the client

loggingestion List of logins used to connect to the application, when the account is managed by the application itself. This table also contains the accounts authorized to use the web services.

Column name	Type	Not null	Key	Comment
id	integer	X	X	
login	character varying	X		Login used by the user
password	character varying			
nom	character varying			Name of the user
prenom	character varying			Surname
mail	character varying(255)			email. Used to send password loss messages
datemodif	timestamp without time zone			Last date of change of the record
actif	smallint			If 1, the account is active and can be logging to the application
is_clientws	boolean			True if the login is used by a third-party application to call a web-service
tokenws	character varying			Identification token used for the third-parties applications
is_expired	boolean			If true, the account is expired (password older)

Referenced by id : gacl.passwordlost (id)

passwordlost Password loss tracking table

ANNEXE B. STRUCTURE DE LA BASE DE DONNÉES

Column name	Type	Not null	Key	Comment
passwordlost_id	integer	X	X	
id	integer	X		Logingestion id key
token	character varying	X		Token used to renewal
expiration	timestamp without time zone	X		Expiration date-time of the token
usedate	timestamp without time zone			Used date-time of the to- ken

References id : gacl.logingestion (id)

Bibliographie

- [1] Eric Quinton. Prototypephp, 2016. URL <https://github.com/equinton/prototypephp/tree/bootstrap>.
- [2] smarty. Smarty, php template engine, 2016. URL <http://www.smarty.net>.
- [3] JQuery. Site officiel, 2015. URL <http://jquery.com/>.
- [4] bootstrap. Bootstrap is the most popular html, css, and js framework for developing responsive, mobile first projects on the web, 2016. URL <http://getbootstrap.com>.
- [5] Apache-FOP. The apache fop project, 2016. URL <http://xmlgraphics.apache.org/fop/>.
- [6] OWASP. Application security verification standard (2014), 2014. URL https://www.owasp.org/images/5/58/OWASP_ASVS_Version_2.pdf.
- [7] OWASP. Site institutionnel, 2015. URL <https://www.owasp.org>.
- [8] OWASP. Zed attack proxy project, 2015. URL https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project.
- [9] Free Software Foundation. Gnu affero general public license, 2007. URL <https://www.gnu.org/licenses/agpl.html>.
- [10] APP. Agence pour la protection des programmes, 2016. URL <http://www.app.asso.fr>.
- [11] Eric Quinton. Documentation d'utilisation du framework prototype php, 2016. URL <https://github.com/equinton/prototypephp/blob/bootstrap/database/documentation/prototypephp-documentation.pdf>.
- [12] Cisco. Clamav® is an open source antivirus engine for detecting trojans, viruses, malware and other malicious threats, 2015. URL <http://www.clamav.net/>.
- [13] ArchLinux. Clamav, 2016. URL <https://wiki.archlinux.org/index.php/ClamAV>.
- [14] Mozilla. Mozilla ssl configuration generator, 2016. URL <https://mozilla.github.io/server-side-tls/ssl-config-generator/>.

- [15] ANSSI. Recommandations de sécurité relatives à tls, 2016. URL http://www.ssi.gouv.fr/uploads/2016/09/guide_tls_v1.1.pdf.
- [16] Eric Quinton. Ré-identification par jeton, 2016. URL <http://www.linux-professionnel.net/programmation/php---codes-divers/re-identification-par-jeton>.
- [17] Mike Benoit and Dan Cech. Phpgacl, generic access control lists, 2006. URL <http://phpgacl.sourceforge.net>.
- [18] Justin Ellingwood. How to set up master slave replication on postgresql on an ubuntu 12.04 vps, 2013. URL <https://www.digitalocean.com/community/tutorials/how-to-set-up-master-slave-replication-on-postgresql-on-an-ubuntu-12-04-vps>.
- [19] Greg Reinacker. Zero to postgresql streaming replication in 10 mins, 2013. URL <http://www.rassoc.com/gregr/weblog/2013/02/16/zero-to-postgresql-streaming-replication-in-10-mins>.
- [20] Nils Hamerlinck. Configurer la rÉplication d'un serveur postgresql, 2015. URL <http://connect.ed-diamond.com/GNU-Linux-Magazine/GLMF-184/Configurer-la-replication-d-un-serveur-PostgreSQL>.
- [21] postgresql. Binary replication tutorial, 2015. URL https://wiki.postgresql.org/wiki/Binary_Replication_Tutorial.
- [22] pgAdmin. pgadmin postgresql tools, 2016. URL <https://pgadmin.org>.
- [23] Eric Quinton. Php - utiliser clamav pour rechercher les virus dans les documents téléversés, 2016. URL <http://www.linux-professionnel.net/programmation/php---codes-divers/php---utiliser-clamav-pour-rechercher-les-virus-dans-les-documents-teliverses>.
- [24] Stanislas Lange. Installer php 7 sous debian 8 jessie via le dépôt dotdeb, 2016. URL <https://angristan.fr/installer-php-7-debian-8-jessie-depot-dotdeb/>.


EABX – Écosystèmes aquatiques et changements globaux
50, avenue de Verdun
33612 CESTAS Cedex
+33 (0)5 57 89 08 00

Rejoignez-nous sur :



**Institut national de recherche pour
l'agriculture, l'alimentation et l'environnement**