Module Réglementaire DSN

DSN 4.1

Moteur déclaratif - Guide Technique

HRa Suite 7 et 9

F REGDSN 2017 1 GT 041 HRA Edition : **1**

Date : **Mai 2017**

Avertissement

Le présent document peut contenir des informations ou des références concernant certains produits, logiciels ou services HR Access non annoncés dans ce pays. Cela ne signifie pas que Sopra HR Software ait l'intention de les y annoncer. Sopra HR Software peut détenir des brevets ou des demandes de brevet couvrant les produits mentionnés dans le présent document. La remise de ce document ne vous donne aucun droit de licence sur ces brevets ou demandes de brevet.
Les logiciels tierce partie inclus dans HR Access ne peuvent être utilisés séparément de HR Access.

Marques

HRa Suite et HR Access sont des marques déposées de Sopra HR Software. Toute utilisation, reproduction ou représentation nécessite l'accord express et préalable de Sopra HR Software. Les autres noms utilisés pour désigner des sociétés, des produits ou des services sont des marques ayant leur titulaire respectif.

Mise à jour de la documentation

Le contenu de la documentation HR Access est régulièrement mis à jour à travers les Delivery Pack

Remarques du lecteur

Vos commentaires et suggestions nous permettent d'améliorer la qualité de nos documentations. Ils jouent un rôle important lors de leur mise à jour. N'hésitez pas à en faire part à la hot-line HR Access.

www.soprahr.com
Le Triangle de l'Arche
8, cours du Triangle
92937 Paris La Défense Cedex

Table des matières

A propos de ce document

Usage

Ce document est le **Guide technique de** l'*application* **Moteur Déclaratif.**

Pour rappel, cette application fait partie du Packaged Module "**Module Réglementaire DSN**", qui contient d'autres éléments tels que le *Rechargeur*.

Ce Rechargeur fait l'objet d'un Guide technique dédié du même type que celui-ci.

Le Guide de Référence...

Le *Guide de Référence* - livré aussi avec votre Delivery Pack - décrit, lui, l'*ensemble* du Module Réglementaire DSN.

Public visé

Ce document s'adresse à des acteurs de type :

- **Administrateur Système et Base de Données**, tout particulièrement pour les opérations d'installation et de réinstallation de l'application.
- **Exploitant Technique**, en particulier pour tous les éléments décrits dans le chapitre "Exploitation".

Cette application est un SERVEUR

Si vous n'êtes pas un acteur des catégories ci-dessus, nous vous conseillons **de vous faire assister** par ces experts dans vos opérations.

Ce document est en effet destiné à ce public et exige donc des connaissances techniques spécifiques pour être appréhendé.

Par ailleurs, en tant que module **serveur,** cette application n'est pas destinée (sauf opérations de test limitées et ponctuelles) à être installée sur un poste client et utilisée comme telle dans cette configuration.

Moteur Déclaratif vs. Espace DSN

Si vous avez souscrit pour l'option **Espace DSN**, un Guide Technique dédié vous sera fourni à sa livraison.

Architecture Technique et prérequis

Référence

Se référer au **Document d'Architecture Technique**.

Points d'attention

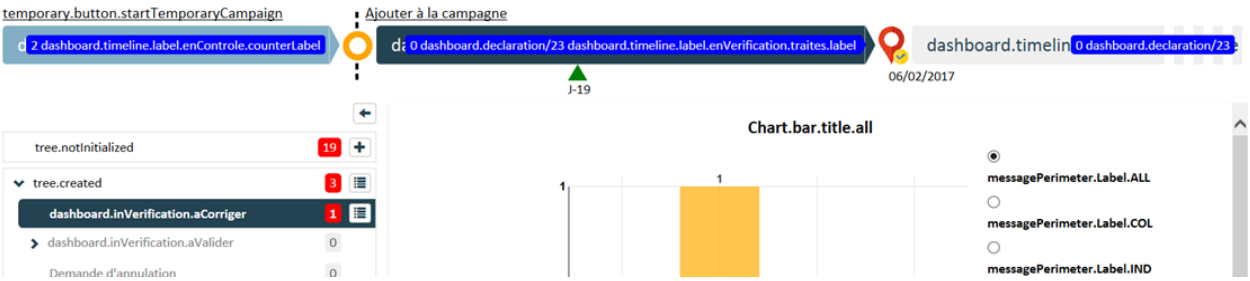
Consignes sur Internet Explorer pour la DSN 4.1

Certains comportements anormaux peuvent être observés sur Internet Explorer au passage DSN 3.3 => DSN 4.1 tels que :

- le bandeau de changement de période qui n'apparaît pas,
- les onglets du haut de la page qui ne fonctionnent pas,
- les libellés qui ne sont pas interprétés.

Pour remédier à ce dysfonctionnement, il convient de **vider manuellement le cache d'Internet Explorer**.

Un exemple d'anomalies affichées à l'écran est donné ci-dessous :



Usage simultané 3.3 / 4.1

Si vous utilisez en **simultané la DSN 3.3 et la DSN 4.1** sur votre Internet Explorer, vous pouvez avoir à refaire la manipulation du vidage du cache décrite ci-dessus ...

Rappel sur les prérequis

Driver JDBC

L'usage de la fonctionnalité **M2M** exige **un driver JDBC supportant la gestion de BLOBs.**

Exemple : JDBC 4.0 pour DB2 V10.5

Java

Niveau de JRE

Note Importante

Comme indiqué dans le document d'architecture technique, le niveau de Java requis est **JRE 1.7**.

Variable JAVA_HOME

Variable d'environnement JAVA_HOME

Au-delà de l'installation du JRE 1.7, vérifier que la **variable d'environnement JAVA_HOME est correctement positionnée** et pointe bien sur le JRE 1.7 lorsque l'application est utilisée.

Bugs connus

JRE IBM

AIX / JRE IBM

La version **initiale** de la série 1.7 d'IBM comporte des **bugs bloquants.**

Pour un fonctionnement correct de l'application, un JRE 1.7 - **SR1** minimum est requis.

JRE et driver JDBC

Certains drivers JDBC anciens peuvent engendrer des problèmes au démarrage si le paramètre datasource.pool.validationQuery n'est pas renseigné.

Voir "**Configuration / Connexions JDBC / datasource.pool.validationQuery**".

Informations générales

Moteur Déclaratif : module, composants, versions

Principes généraux

Ce chapitre donne les éléments généraux sur le **Moteur Déclaratif,** concernant ses packages de distribution, leurs éléments constitutifs, leurs structures et leurs évolutions dans les différentes versions qui vous sont livrées.

Il vous permet d'appréhender le guide avec une connaissance plus globale et vous en facilite ainsi la lecture.

Module Réglementaire, Moteur Déclaratif, application, composants, numérotations

Le Module de livraison

Le Moteur Déclaratif vous est livré au sein du **Module réglementaire DSN** qui est **un assemblage d'applications et de composants.** Vous y trouverez par exemple :

- l'application *Moteur Déclaratif* qui est l'objet même de ce guide
- l'application *Rechargeur*
- leur documentation
- **…**

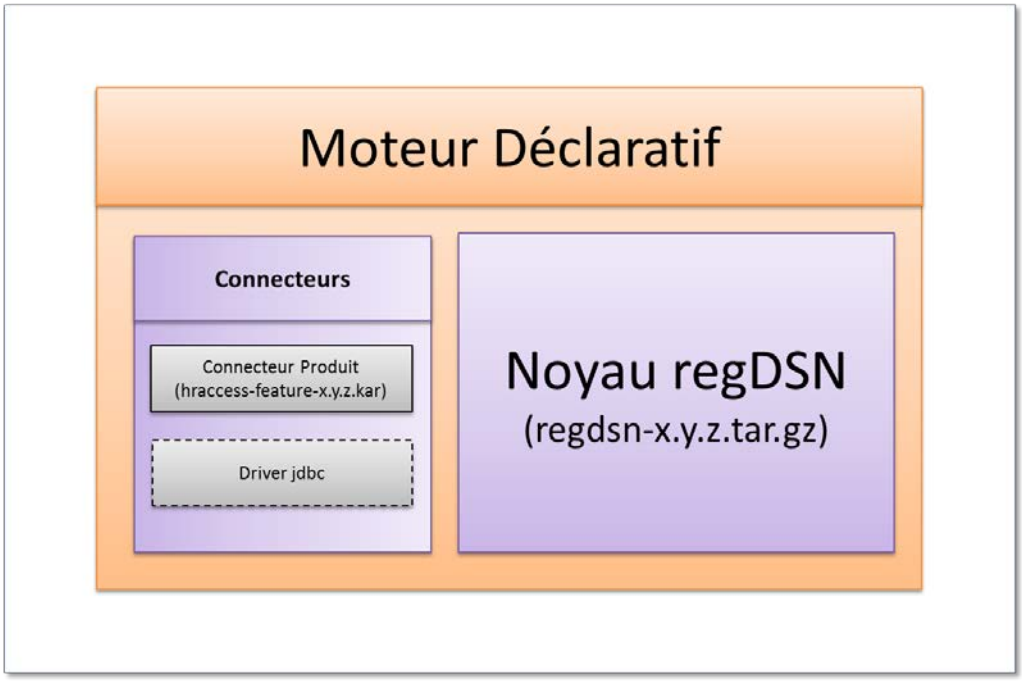
Le Moteur Déclaratif

Le Moteur Déclaratif procède d'un assemblage de composants dont les principaux sont :

- le **noyau de l'application**, reg**DSN**
  - livré sous la forme d'un fichier compressé de type *tar.gz*. regdsn-x.y.z.tar.gz

- des **connecteurs**, autrement appelés **add-ons**, comme :
  - le **connecteur Produit**, livré sous forme d'un fichier .kar, et nécessaire à l'accès *applicatif* aux données
  - le **driver JDBC** nécessaire à l'accès *technique* aux données

D'autres éléments, comme *la brique de contrôle DSN-Val*, y sont aussi intégrés



Chacun de ses éléments, du module au composant, possède sa nomenclature et sa numérotation.

Exemple

A titre d'exemple, le *Module Réglementaire DSN* **3.3.18** intègre un Moteur Déclaratif constitué :

- d'un *noyau regDSN* de version **3.7.5**
- d'un *connecteur HR Access* de version **3.3.5**

L'exemple ci-dessus montre clairement que les composants contenus dans le module 3.3.18 (ici le noyau et le connecteur) **ne portent pas le même numéro que le module conteneur**.

Ceci est **totalement normal** et traduit l'évolution unitaire de ces sous-éléments indépendamment de leur assemblage.

Les versions et leurs usages

Version externe

La communication sur les versions DSN qui vous est faite utilise un numéro de version **externe** de **niveau Module** (le plus haut).

```
Exemple: Module Réglementaire DSN - 3.3.10
```

Ce numéro de version externe est utilisé pour identifier le **Delivery Pack** (fichier zip) qui vous est fourni pour livraison du module :

```
Ex: REGDSN-3.3.10.zip
```

Ce numéro de version externe est utilisé dans ce guide pour identifier les **fonctionnalités** liées à un **niveau de version donné**.

C'est le cas par exemple des **fonctionnalités des commandes** documentées dans le chapitre "Référence des commandes" : vous trouverez ainsi des précisions du type "A partir de la 3.3.10", pour les commandes ou certaines de leurs options.

Explorer les nouveautés fonctionnelles

Vous pouvez faire une recherche dans ce guide en utilisant les numéros de version des Delivery Pack pour connaître le détail des nouveautés par version.

Versions internes

Les versions internes sont celles des composants et applications contenus dans le Delivery Pack.

Les plus importantes d'entre elles sont :

- la version du **noyau regDSN**
- la version du **connecteur** HR Access

Version du noyau regDSN

Vous pouvez le visualiser :

- sur l'archive tar.gz de l'application dans le .zip qui vous a été fourni.

```
Ex : regdsn-2.2.2.tar.gz
```

- au lancement du Moteur Déclaratif en mode interactif.

Le numéro de **version du noyau regDSN** s'affiche sous son logo.

- sur le chemin du répertoire d'installation.

Version du connecteur

Vous pouvez le visualiser :

- sur l'archive .kar qui vous est livrée.

```
Ex : hraccess-feature-3.3.5.kar
```

- lorsque vous exécutez la commande addons:list.

Le numéro de **version du connecteur** s'affiche derrière son nom (hraccess-feature-**x.y.z**.kar).

```
D:\regdsn-2.1.0\bin>client.bat "addon:list"
Logging in as regdsn
Name               | Version                  | Type | Full name
-------------------------------------------------------------------------
hraccess-feature   | 3.3.5                    | kar  | hraccess-feature-
2.0.5.kar
oracle-jdbc-driver | 11.1.0.7.0-Production    | jar  | oracle-jdbc-driver-
11.1.0.7.0-Production.jar
```

Usages

Ces versions internes pourront vous être demandées par les équipes Support pour vérifier la configuration installée.

Au sein de ce guide, la **version du noyau regDSN** est un élément **clef** des évolutions du Moteur Déclaratif : son changement traduit des évolutions structurelles, qui conditionnent la procédure de "**Mise à jour d'une version installée**" documentée dans

le chapitre "**Installation**". Le paragraphe "**Versions et modifications structurelles**" ci-dessous vous en détaille les raisons.

Versions et modifications structurelles

La procédure d'installation initiale décrite dans ce guide vous permet une première installation complète de l'application, qui doit avoir lieu au moins une fois sur chaque environnement.

Mais après cette installation initiale, la mise en place d'une nouvelle version s'opère généralement en **mise à jour de l'installation existante**, via une procédure également décrite dans ce guide. C'est en particulier indispensable sur les **environnements de production**.

Cette mise à jour doit alors veiller à prendre en charge les modifications structurelles potentiellement présentes dans la nouvelle version.

Ce paragraphe vous permet d'en comprendre les principes et modes de mise en œuvre.

Première acquisition

Si vous lisez ce guide pour une première installation, vous pourrez lire le chapitre ci-dessous ultérieurement.

Règles générales

Les versions du Module Réglementaire DSN ont une numérotation du type - Module Réglementaire DSN - x.y.zz. Elles sont de deux types :

- **versions fonctionnelles** : elles se repèrent par un changement sur le 2ème chiffre de leur numérotation
  - Ex : DSN-**4.1** vs. DSN-**3.3**
- **versions de maintenance** : elles se repèrent par un changement sur la 3ème partie de leur numérotation
  - Ex : DSN-3.1.**30** vs. DSN-3.1.**20**

Les changements de versions peuvent induire **des changements structurels.**

Ces changements structurels sont **essentiellement des modifications de la structure de données BDSN :**

- Les versions **fonctionnelles** incluent quasi-**systématiquement** des changements structurels.

- Dans les versions **de maintenance,** ils sont *moins fréquents***.** Cependant, ils ***peuvent* avoir lieu** pour mettre en place des correctifs de bas-niveau, ou pour déployer des fonctionnalités additionnelles réputées obligatoires sur les versions en production.

Les modifications structurelles sont toujours liées à un changement de version **du noyau regDSN sur ces deux premiers chiffres.**

Exemple : regdsn-**4.0** vs. regdsn-**3.3**

Modifications structurelles et versions

Quelle que soit la nature de la version, les modifications structurelles sont donc **repérables** par le **changement de la version du noyau regDSN sur ses deux premiers chiffres.**

Corollaire : Si la version du noyau change sur ses deux premiers chiffres, vous avez des modifications structurelles à prendre en compte.

Nature des modifications structurelles

Les modifications structurelles sont principalement des **mises à jour de la BDSN :** elles sont décrites et livrées sous forme de **scripts d'update**.

Certains éléments de **configuration** peuvent aussi être décrits dans les changements structurels**, en tant que paramétrage additionnel** à opérer sur les nouveautés fournies. Ils *ne sont* en général *pas structurants* dans la mesure où les paramétrages proposés possèdent des valeurs par défaut qui les rendent opérationnels et compatibles.

Ces éléments sont décrits sous forme de références aux paragraphes du chapitre "Configuration" de ce guide. Ils sont cités comme des points d'information et d'attention.

Scripts d'update : nomenclature et mode d'emploi

Les scripts d'update sont stockés dans le répertoire **ddl/update** de l'archive qui vous est livrée.

Nature

Ils concernent la mise à jour des **tables** de données des **déclarations**.

Ils font tous référence aux **versions du noyau regDSN** entre lesquelles les changements structurels ont eu lieu.

C'est en cela que la version du noyau a un rôle clef dans les transitions.

Nomenclature

Ils ont une nomenclature du type :

```
edsn.[SGBD].update_[vFROM]_[vTO].sql
```

NB : [SGBD] désigne le type de base de données que vous utilisez, c'est-à-dire : oracle, mssql, db2-zos, db2-os400, db2-aix

vTO et **vFROM** désignent des versions du **noyau regDSN.**

Pour rappel, la version du noyau regDSN est visible :

- sur l'archive tar.gz fournie (regdsn-**x.y.z**.tar.gz)
- au lancement du Moteur Déclaratif

Pour prendre en compte les modifications structurelles de BDSN et appliquer les scripts d'update associés, il vous faut :

1. Identifier la **version courante du noyau**, **vSource**, repérable sur **le nom du répertoire d'installation** ou en **lançant un shell client** (la version s'affiche sous le logo).

2. Identifier le numéro de **la version du noyau** que vous vous apprêtez **à installer**, **vTarget**, repérable sur **le nom de l'archive tar.gz** livrée.

3. Appliquer en séquence croissante **tous les scripts d'update** pour lesquels **vFrom >= vSource**, et **vTo <= vTarget**.

Le paragraphe ci-dessous est là pour **garder trace** de l'ensemble des **changements structurels** intervenus au cours des versions produites.

En effet, lorsque vous changez de version DSN, il vous faut appliquer **tous les changements qui ont eu lieu** entre la version que vous **possédez** et celle que vous **installez**.

Des exemples de transition vous sont donnés pour comprendre l'usage du tableau d'historique.

Historique des changements structurels

Tableau récapitulatif

Lecture	Version du noyau eDSN	Nature du changement	Elément(s) de remise à niveau
^ | | | | | |	4.0	Structure de BDSN - schéma tables *Déclarations*	edsn.[SGBD].update_3.7.x_4.0.x.sql
	3.7	Structure de BDSN - schéma tables *Déclarations*	edsn.[SGBD].update_3.6.x_3.7.x.sql

(*) Remarque pour les plateformes DB2 et les script "edsn.db2-*.update_3.3.x_3.4.x.sql":

Une erreur s'affichera lors de l'exécution du script concernant une commande 'drop index' sur l'un des 2 indexes

- 'IXdecIdRemuneration' non existant si mise à jour depuis un edsn installé en 3.x

- 'IXdecIdRemuneratio' non existant si mise à jour depuis un edsn installé originellement en 2.x

Veuillez ne pas tenir compte de l'erreur.

Si seule cette erreur est présente, l'exécution du script de mise à jour sera à considérer comme réussie.

Exemples

Exemples

J'installe le Moteur Déclaratif du **Module Réglementaire DSN 2.0.70**, c'est-à-dire une version cible du noyau regDSN égale à **2.2.x**...

Cas 1 : j'avais au préalable un **Module Réglementaire DSN 1.1.40**, c'est-à-dire une version source du noyau regDSN égale à **1.1.x**.

j'appliquerai en séquence :

- **edsn.*SGBD*.update_1.1.x_2.0.x.sql**

puis

- **edsn.*SGBD*.update_2.1.x_2.2.x.sql**

Cas 2 : j'avais au préalable un **Module Réglementaire DSN 2.0.60**, c'est-à-dire une version source du noyau regDSN égale à **2.1.x**.

j'appliquerai :

- **edsn.*SGBD*.update_2.1.x_2.2.x.sql**

Signalements

Principes généraux

Evénements de gestion

Les signalements sont issus *d'événements de gestion* produits au niveau de la Gestion Administrative (Arrêt de travail ou Reprise anticipée) et de la Paie (Fin de contrat).

Ils donnent lieu à la production de *DSN de Signalement* qui sont gérées au niveau du Moteur déclaratif.

Processing de événements

Ces événements, une fois *émis* ou *modifiés*, doivent être **traités** pour donner lieu à la production de DSN de Signalement dans la BDSN : lors du lancement du traitement, appelé *processing des événements*, une déclaration de Signalement est ajoutée si l'historique des DSN mensuelles précédentes l'autorise.

Une fois traité, l'événement ne sera plus pris en compte lors du lancement d'un processing ultérieur.

Modification et événements

Lorsqu'un événement de gestion donné est *modifié* (Exemple : modification du motif d'arrêt maladie), il sera de nouveau pris en compte dans le processing des événements.

Production des flux de Signalement

Une fois les DSN de Signalement créées, elles devront elles-mêmes être traitées pour donner lieu aux flux de signalement correspondants, de type *Initial*, *Annule et Remplace* ou *Annulation*.

Ce traitement est effectué :

- **unitairement** sur la DSN de Signalement si le signalement est de type **Initial**.
- par **couples de DSN** pour les autres cas.

Commandes dédiées

Dans cette version, une nouvelle série de commandes spécifiques a été introduite pour procéder au processing des événements de gestion ou à la production des flux DSN de Signalement associés :

- elles sont préfixées par **`dsnsig:`**
- elles sont documentées dans le chapitre "**Exploitation / Référence des commandes**"

Installation

Principes généraux

Composants, répertoires, et base de données

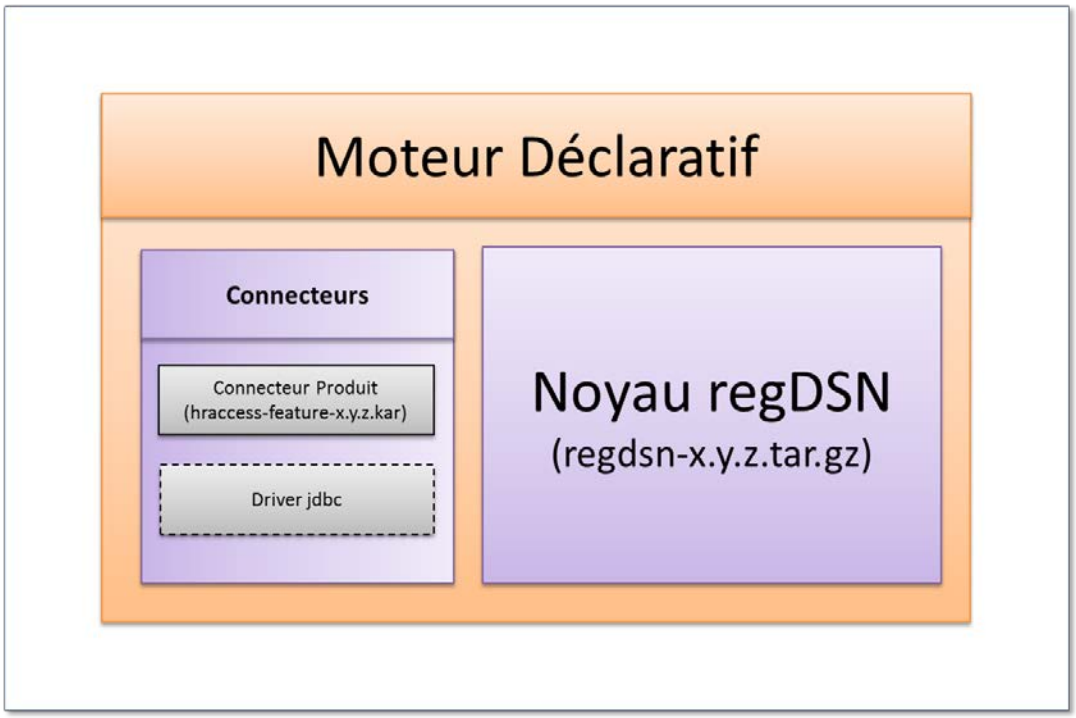
Pour rappel et tel que décrit dans le chapitre " **Informations générales**" :

Le Moteur Déclaratif procède d'un assemblage de composants dont les principaux sont :

- le **noyau de l'application**, reg**DSN**
  - livré sous la forme d'un fichier compressé de type *tar.gz*. regdsn-x.y.z.tar.gz

- des **connecteurs**, autrement appelés **add-ons**, comme :
  - le **connecteur Produit**, livré sous forme d'un fichier .kar, et nécessaire à l'accès *applicatif* aux données
  - le **driver JDBC** nécessaire à l'accès *technique* aux données

D'autres éléments, comme *la brique de contrôle DSN-Val*, y sont aussi intégrés.



D'autre part, l'application :

- nécessite un espace de persistance SGBD dédié : la **BDSN**.
- gère sa **configuration** dans un répertoire dédié, qui est **conservé d'une installation à l'autre**.

Répertoire de configuration

Ce répertoire **persistant** s'appelle par défaut **regdsn-home**.

Les connecteurs (ou add-ons) servent à la **connexion avec les bases de données** : BDSN et base HR Access.

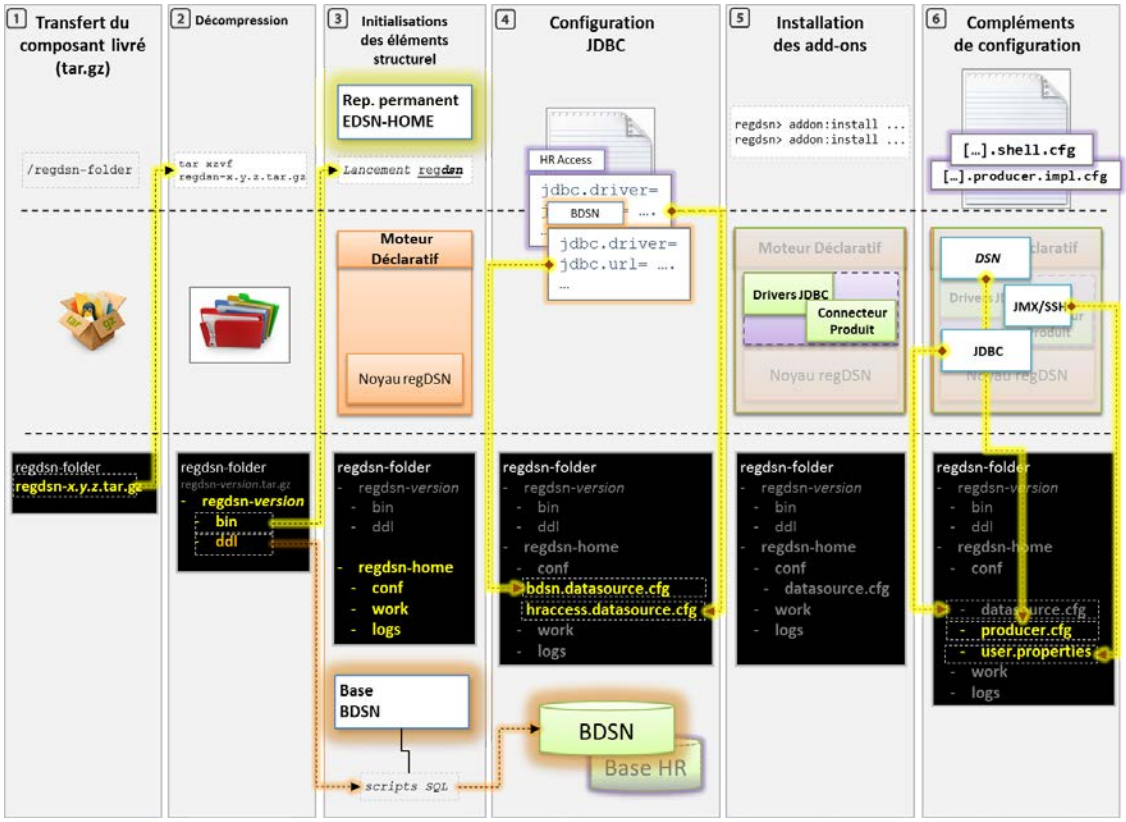
L'installation consiste :

- à *mettre en place* l'ensemble de ces composants.
- à *configurer* ces éléments.

Le schéma ci-dessous résume les principales étapes de ce processus.

A noter que :

- ce schéma n'a pas vocation à être exhaustif : les éléments figurés sont illustratifs.
- les *commandes systèmes* présentées (telles que celle de décompression de l'archive tar.gz) peuvent varier en fonction du système d'installation.



Procédures d'installation

Les différents cas de figure liés à l'installation concernent :

- l'installation initiale
- la mise à jour d'une version déjà installée
- l'installation sur un autre environnement

Les chapitres suivants vous décrivent les modalités de ces différentes procédures d'installation.

Installation initiale

Mise en place de l'archive

Transfert de l'archive

Penser à transférer l'archive en mode **binaire.**

L'archive se nomme regdsn-*version*.tar.gz.

La première étape consiste à :

- **transférer** ce livrable sur la machine où le Moteur Déclaratif doit être installé, et ce, dans le répertoire de votre choix.
- **décompresser** ce tar.gz.

Concernant la décompression :

- Sous **Windows**

Fichiers tar sous Windows

Sous Windows on peut utiliser l'outil gratuit 7-Zip pour décompresser le livrable.

- Sous **Unix**

Il existe des utilitaires tar natifs sur ces systèmes.

Erreurs de décompression

Il arrive que la décompression de l'archive génère des erreurs.

Parmi les erreurs fréquemment rencontrées, l'apparition de messages de type **@LongLink.**

Dans la grande majorité des cas, ces erreurs sont liées aux **utilitaires tar natifs** de votre système d'exploitation.

Pour remédier à ces erreurs, utilisez une implémentation **GNU** de l'utilitaire (**GNU tar**) que vous pouvez récupérer librement.

Ceci créera un répertoire **regdsn-*version*** dans lequel on trouve :

- un répertoire **bin** contenant les scripts de démarrage et d'arrêt du Moteur Déclaratif
- un répertoire **ddl** contenant les scripts SQL d'initialisation de la base BDSN
- d'autres répertoires *privés*, indispensables au fonctionnement interne du Moteur Déclaratif et qu'il ne faut pas modifier.

Initialisations structurelles

Les **éléments structurels** traités dans ce chapitre désignent les éléments qui seront conservés d'une installation à l'autre. Il s'agit du répertoire persistant **regdsn-home** et de la **BDSN**.

Le répertoire persistant est automatiquement créé s'il n'existe pas ; la BDSN nécessite d'être créée manuellement.

Initialisation du répertoire de persistance

Un premier démarrage du Moteur Déclaratif est nécessaire pour initialiser le répertoire persistant.

Par défaut, ce répertoire se nommera **regdsn-home** et se trouvera dans le même répertoire que le répertoire d'installation du Moteur Déclaratif. Par exemple, si le moteur déclaratif est installé dans **/opt/regdsn-1.0.0**, le répertoire persistant se trouvera dans **/opt/regdsn-home**.

Changer regdsn-home

Il est possible de **modifier l'emplacement du répertoire persistant** en alimentant la variable d'environnement **REGDSN_HOME**.

Pour démarrer le Moteur Déclaratif, allez dans le répertoire bin et exécutez le script **regdsn.bat** (sous Windows) ou **regdsn** (sous Unix).

JRE 1.7 & JAVA_HOME

Penser à ce que le JAVA_HOME utilisé par les shells de lancement soit **celui du JDK/JRE 1.7** dans le cas où *plusieurs JDKs sont installés*.

Cela donne accès au shell du Moteur Déclaratif :

```
                    ___    _____ _    _
   _________  ____ _/ __ \/ ___// | / /
  / ___/ _ \/ __ `/ / / /\__ \/  |/ /
 / /  /  __/ /_/ / /_/ /___/ / /|  /
/_/   \___/\__, /_____//____/_/ |_/
          /____/

Moteur declaratif (3.7.5)

Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit '<ctrl-d>' or type 'system:shutdown' or 'logout' to shutdown engine.

regdsn>
```

Une fois le shell disponible, arrêtez le Moteur Déclaratif en tapant :

```
regdsn>system:shutdown
```

Le répertoire persistant est maintenant créé. Il contient :

- Un répertoire **conf** : on y retrouve tous les fichiers de configuration du Moteur Déclaratif
- Un répertoire **logs** : c'est ici que seront produites les logs du Moteur Déclaratif
- Un répertoire **work** : c'est un répertoire de travail nécessaire au bon fonctionnement du Moteur Déclaratif

Initialisation de la BDSN

La BDSN

La BDSN désigne l'ensemble des tables requises pour la gestion des déclarations.

Ces tables sont **nouvelles** et définissent un espace de base de données propre, en complément des tables Produit existantes.

L'ensemble des explications ci-dessous donne les instructions pour initialiser la BDSN.

A noter que si la structure de cette base est amenée à changer dans les versions de maintenance ou dans les versions fonctionnelles, ce changement vous sera indiqué dans les Delivery Pack et les instructions de remise à niveau vous seront fournies.

Les fichiers DDLs

La BDSN peut être installée sur Oracle, DB2 ou SQL Server. Les scripts SQL d'initialisation de la base de données se trouvent dans le répertoire ddl/create. On y trouve un répertoire par type de base :

- **db2-aix** pour les bases de données IBM DB2 AIX
- **db2-os400** pour les bases de données IBM DB2 AS400
- **db2-zos** pour les bases de données IBM DB2 zOS
- **mssql** pour les bases de données Microsoft SQL Server
- **oracle** pour les bases de données Oracle

DDLs PostgreSQL

Vous trouverez dans le package des DDLs pour la base de données PostgreSQL. Ces DDLs sont livrées à titre illustratif et ne sont pas maintenues par HR Access.

Il est nécessaire d'exécuter tous les scripts SQL présents dans le répertoire correspondant au type de base.

Des **actions préalables** sont nécessaires avant d'exécuter ces scripts. Ces actions sont spécifiques au type de la base de données utilisée.

Oracle

- **DDLs et Tablespaces**

Le fichier ddl fourni (edsn.oracle.create.sql) crée des objets dans des tablespaces dédiés. Il est nécessaire de créer au préalable un* tablespace pour les données* et un **pour les indexes**.

- **Création des Tablespaces**

Les deux tablespaces à créer sont décrits dans le fichier 'bdsn.tablespace.oracle.create.sql' :

```
CREATE TABLESPACE BDSN_DATA DATAFILE
'/instanceOracle/oradata/BDSN/data/BDSN_D01.dbf' SIZE 500M REUSE AUTOEXTEND
ON NEXT 100M MAXSIZE 10G;
CREATE TABLESPACE BDSN_INDEX DATAFILE
'/instanceOracle/oradata/BDSN/data/BDSN_I01.dbf' SIZE 500M REUSE AUTOEXTEND
ON NEXT 100M MAXSIZE 10G;
```

Dans l'exemple ci-dessus :

- Le nom de la base est **BDSN**.
- Le nom du tablespace des données est **BDSN_DATA** et son fichier associé est **BDSN_D01.dbf**.

- Le nom du tablespace des indexes est **BDSN_INDEX** et son fichier associé est **BDSN_I01.dbf**.
- Le chemin (path) de l'installation Oracle est générique, vous devez le modifier en fonction de votre installation.
- Les deux tablespaces sont créés avec une taille initiale de 500M et maximale de 10G, par incrément d'augmentation de 100M. La taille des tablespaces devra être ajustée en fonction de votre contexte d'utilisation.

Noms de tablespaces personnalisés

Si vous souhaitez changer le nom des tablespaces, veuillez noter que vous **devrez modifier manuellement les fichiers ddl** , afin de reporter ce changement. Par défaut, le fichier ddl Oracle fourni se base sur ces deux noms de tablespaces : BDSN_DATA et BDSN_INDEX.

DB2 aix

- **DDLs et Tablespaces**

Le fichier ddl fourni (edsn.db2-aix.create.sql) crée les objets dans des tablespaces dédiés et un tablespace standard (USER).

Il est nécessaire de créer au préalable un certain nombre d'objets db2 (bufferpool, tablespaces), comme décrit dans le paragraphe ci-dessous.

- **Création des objets db2**

Il est préconisé d'exécuter les commandes suivantes, comme décrit dans le fichier **bdsn.tablespace.db2-aix.create.sql**

```
CREATE BUFFERPOOL BUFHR32K ALL NODES SIZE 1000 PAGESIZE 32768;
CREATE TABLESPACE USER32K PAGESIZE 32768 MANAGED BY SYSTEM USING
('/instanceDB2/database/data32k') BUFFERPOOL BUFHR32K;
CREATE TEMPORARY TABLESPACE TEMPORARY32K PAGESIZE 32768 MANAGED BY SYSTEM
USING ('/instanceDB2/database/sys/TEMP32K') BUFFERPOOL BUFHR32K;
CREATE REGULAR TABLESPACE "TSDSN" IN $DATABASE PARTITION GROUP
IBMDEFAULTGROUP PAGESIZE 4096 MANAGED BY SYSTEM
     USING ('/instanceDB2/database/data4k/TSDSN')
     EXTENTSIZE 32
     PREFETCHSIZE AUTOMATIC
     BUFFERPOOL IBMDEFAULTBP
     OVERHEAD 7.500000
     TRANSFERRATE 0.060000
     NO FILE SYSTEM CACHING
     DROPPED TABLE RECOVERY ON;
```

Dans l'exemple ci-dessus :

- Le nom du tablespace dédié est **TSDSN**.
- Le nom du tablespace 32k est **USER32K**.
- Le nom du bufferpool est **BUFHR32K**.
- Le nom du tablespace temporaire est **TEMPORARY32K**.
- Le chemin (path) de l'installation db2 est générique, vous devez le modifier en fonction de votre installation.

Objets DB2 personnalisés

Si vous souhaitez changer le nom des tablespaces, veuillez noter que vous devrez **modifier manuellement les fichiers ddl**, afin de reporter ce changement. Par défaut, les fichiers ddl DB2 aix fournis se basent sur ces noms de tablespaces : **USER**, **USER32K** et **TSDSN**.

De plus, les fichiers ddl fournis utilisent les variables suivantes :

- Le **nom de la database**, défini par la variable **$DATABASE**.
- Le **nom du schéma**, défini par la variable **$OWNER**.

Il sera indispensable de les remplacer avant leurs exécutions.

DB2 os400

Le fichier ddl DB2 fourni (edsn.db2-os400.create.sql) crée les objets dans la collection choisie, précisée par un radical dédié.

Dans ce fichier SQL à exécuter, il est nécessaire de **remplacer le nom du radical, défini par la variable $OWNER.**

La collection pourra être créée ainsi (si non existante ou si souhait d'une collection spécifique pour la BDSN) :

```
create collection $OWNER
```

Pour cette base de données, il existe des directives particulières pour la configuration JDBC qui seront détaillées ultérieurement.

DB2 zOS

- **DDLs, qualifier et STDGROUP**

Le fichier ddl fourni (edsn.db2-zos.create.sql) crée les objets dans la database choisie, pour un **qualifier** dédié et un **STOGROUP**.

Le paragraphe suivant explique comment créer ces objets.

- **Création des objets DB2**

Dans les fichiers SQL à exécuter, il est nécessaire de remplacer :

- Le nom de la **database**, défini par la variable **$DATABASE**.
- Le nom du **qualifier**, défini par la variable **$QUALIFIER**.
- Le nom du **STOGROUP**, défini par la variable **$STOGROUP**.

Il est aussi à noter que les scripts de création des objets utilisent deux **bufferpool** standard :

- **BP0**
- **BP32K**

Bufferpools personnalisés

Si, pour votre site, les noms de bufferpool sont différents, vous devrez aussi les modifier dans les fichiers ddl fournis.

Pour la configuration JDBC à la base de données, voir **Configuration des connexions aux bases de données**, paragraphe *DB2 zOS*. Il existe en effet des compléments spécifiques à cette configuration.

SQL Server

Le fichier ddl fourni (edsn.mssql.create.sql) crée des objets dans la database MS SQL Server choisie.

Ce fichier peut être exécuté sans prérequis préalable ni contrainte particulière.

Configuration des connexions JDBC aux bases de données

Principes généraux

Il faut à présent configurer la connexion JDBC à la BDSN ainsi que la connexion JDBC à la base de données HR Access.

Pour cela, **créez** les fichiers suivants dans le répertoire **xxx-home/conf** :

- Fichier **com.soprahr.edsn.datasource.cfg** pour la connexion *à la BDSN*
- Fichier **com.soprahr.edsn.hraccess.datasource.cfg** pour la connexion *à la base de données HR Access*

Par ailleurs, si le préfixe des tables HR Access **n'est pas HR**, il conviendra également de créer dans **xxx-home/conf** le fichier :

- **com.soprahr.edsn.hraccess.connector.cfg**

Préfixe spécifique des tables HR

Pour la configuration d'un préfixe spécifique, voir le chapitre **Configuration/Paramétrage du connecteur HR Access** , paramètre **extension.tablePrefix**

Avertissement et recommandations

Fichiers de configuration

Les fichiers de configuration du Moteur Déclaratif se trouvent dans le répertoire **regdsn-home/conf**.

L'ensemble de ces fichiers est nécessaire au bon fonctionnement du Moteur Déclaratif, mais **seuls les fichiers décrits ci-dessous sont modifiables**. Ils sont classés par thème

Les autres fichiers de configurations que ceux cités ici ...

Les autres fichiers doivent être laissés intacts.

Rappel préalable sur les fichiers de configuration

Les fichiers de configuration suivent les règles générales suivantes :

> S'ils n'existent pas dans votre répertoire conf/, ils doivent être créés manuellement.

Ces fichiers de configuration sont préservés d'une installation à l'autre : le processus d'installation ne les écrase pas. Ce mécanisme explique que lors de la première installation, **ces fichiers peuvent ne pas exister**. Il conviendra alors de les créer lorsqu'ils seront configurés.

> Leur format de stockage est le format ANSI.

Ils ne doivent pas être enregistrés au format UTF-8, car ils ne seraient alors plus lisibles et engendreraient des anomalies graves de fonctionnement.

Test externe des connexions JDBC

Pour tester la validité de votre configuration JDBC, il est *recommandé* de vérifier le paramétrage et les drivers JDBC que vous comptez utiliser **à l'aide d'outils externes**.

Parmi l'ensemble des éléments de configuration, ceux de la configuration JDBC peuvent en effet être contrôlés de manière **isolée et indépendante** avec ces outils : en cas d'anomalie JDBC, le diagnostic et la mise au point seront plus rapides par ce biais.

Test des connexions JDBC

Pour vos tests, vous pouvez par exemple utiliser **DbVisualizer**.

Configuration

Chaque fichier doit contenir les paramètres suivants :

```
# Driver JDBC
datasource.jdbc.driver=

# URL JDBC de la base de données
datasource.jdbc.url=

# Utilisateur de la base de données
datasource.user=

# Mot de passe de la base de données
datasource.password=
```

Renseignez pour chaque connexion les bonnes valeurs de paramètres.

Mots de passe renforcés

Il est possible de **chiffrer les mots de passe** pour ne pas qu'ils apparaissent en clair dans les fichiers de configuration : Voir le paragraphe **Chiffrement des mots de passe**.

Particularités par SGBD

Configuration du connecteur JDBC pour DB2 os400

Dans les deux fichiers de configuration JDBC, il convient de rajouter :

```
;date format=iso;
```

à la **fin du paramètre de l'URL JDBC**.

Un exemple de fichier de configuration est :

```
datasource.jdbc.driver=com.ibm.as400.access.AS400JDBCDriver
datasource.jdbc.url=jdbc:as400:host:port/$OWNER;date format=iso;
datasource.user=user
datasource.password=passwd
```

Configuration hibernate pour DB2 zOS

Il est nécessaire d'ajouter l'élément de configuration suivant dans **conf/bdsn.hibernate.properties** :

```
dialect=org.hibernate.dialect.DB2390Dialect
```

Création initiale du fichier hibernate.properties

Si le fichier **conf/bdsn.hibernate.properties** n'existe pas, il convient de le créer au préalable.

Configuration hibernate pour Oracle 12c

Il est nécessaire d'ajouter l'élément de configuration suivant dans **conf/bdsn.hibernate.properties** :

```
dialect=org.hibernate.dialect.Oracle10gDialect
```

Installation des add-ons

L'étape suivante (installation des add-ons) s'effectue à l'aide du langage de commande du Moteur Déclaratif.

Il convient donc de le **redémarrer**, en exécutant le script **regdsn.bat** (sous Windows) ou **regdsn** (sous Unix).

Les différents types d'add-ons

Les add-ons à installer sont :

- Le **connecteur** permettant d'interagir avec le système HR Access.
- Le ou les **drivers JDBC** permettant de se connecter aux bases de données (BDSN et base HR Access).

Le connecteur

Le **connecteur** se présente sous la forme d'un fichier kar.

Il est livré au sein de la fourniture.

Les drivers JDBC

Les drivers JDBC sont des fichiers jar livrés par le fournisseur de base de données.

Drivers JDBC

Pour les drivers JDBC, procédez de la même manière que pour votre *serveur de Query*.

Commande d'installation

Pour installer un add-on, utilisez la commande **addon:install** depuis le shell du moteur déclaratif en indiquant le chemin vers l'emplacement du fichier jar ou kar.

Par exemple :

```
regdsn>addon:install C:/test/hraccess-feature-1.0.0.kar
regdsn>addon:install D:/Download/oracle-jdbc-driver-11.1.0.7.0-
Production.jar
```

Attention : Même sous Windows, on doit utiliser le caractère '/' comme séparateur de fichier.

Premier test de bon fonctionnement

Tous les éléments d'installation ne sont pas finalisés à ce stade, mais le Moteur Déclaratif est maintenant opérationnel.

Pour le vérifier, tapez la commande :

```
regdsn>bundle:list
```

Vous devez obtenir une liste du type :

```
START LEVEL 100 , List Threshold: 50
ID  | State  | Lvl | Version       | Name
 69 | Active |  80 | 3.0.1         | Apache Karaf :: JNDI :: Command
 75 | Active |  80 | 2.6           | Commons Lang
 76 | Active |  80 | 3.2.1         | Commons Collections
 77 | Active |  80 | 1.6.0         | Commons Pool
```

```
 78 | Active |  80 | 1.4.0.3       | Apache ServiceMix :: Bundles :: commons-dbcp
109 | Active |  80 | 3.2.2.1       | Apache ServiceMix :: Bundles :: mybatis
110 | Active |  80 | 4.11.0.1      | Apache ServiceMix :: Bundles :: junit
111 | Active |  80 | 5.5.0.Final_2 | Apache ServiceMix :: Bundles :: drools
…
```

Statut des bundles

Certains Bundles peuvent **ne pas** apparaître en "Active" à ce stade : votre configuration n'est en effet pas finalisée, certains ports peuvent par exemple être déjà utilisés.

Compléments de configuration

Pour procéder à la finalisation de la configuration, il convient d'arrêter le Moteur Déclaratif en tapant :

```
regdsn>system:shutdown
```

La finalisation de l'installation consiste à configurer l'ensemble des éléments requis pour le bon fonctionnement du Moteur Déclaratif tels que :

- les accès de monitoring et pilotage batch de l'application : chapitre "**Services JMX et SSH**"
- les compléments de configurations JDBC : chapitre "**Configuration JDBC**"
- le paramétrage complémentaire du connecteur HR Access, notamment pour la gestion des préfixes de tables spécifiques : chapitre "**Paramétrage du connecteur HR Access**"
- les paramètres de flux DSN : chapitre "**Paramétrage de la structure S10**"

Etudiez attentivement ces éléments de configuration

Ces chapitres doivent être **étudiés en détail** et mis en œuvre, pour :

- garantir un bon fonctionnement technique de l'application.
- vous permettre de réaliser les **tests de bon fonctionnement de l'installation** décrits dans le *Guide de Référence* de votre Delivery Pack.

Une lecture de l'ensemble est donc nécessaire.

Après la configuration ...

Une fois que vous aurez pris connaissance et mis en œuvre les éléments figurant dans le chapitre " **Configuration**", vous pourrez étudier les **Scripts du Moteur Déclaratif** pour découvrir l'ensemble des scripts disponibles.

Test final de bon fonctionnement

Après avoir finalisé la configuration, vous pouvez valider qu'elle est opérationnelle en tapant de nouveau la commande :

```
regdsn>bundle:list
```

Dans la liste qui s'affiche alors, **tous les bundles doivent être en statut 'Active'**.

Si ce n'est pas le cas, c'est qu'il y a une anomalie dans votre installation ou votre configuration.

```
START LEVEL 100 , List Threshold: 50
ID  | State | Lvl | Version       | Name
69  | Active | 80 | 3.0.1         | Apache Karaf :: JNDI :: Command
75  | Active | 80 | 2.6           | Commons Lang
76  | Active | 80 | 3.2.1         | Commons Collections
77  | Active | 80 | 1.6.0         | Commons Pool
78  | Active | 80 | 1.4.0.3       | Apache ServiceMix :: Bundles :: commons-dbcp
109 | Active | 80 | 3.2.2.1       | Apache ServiceMix :: Bundles :: mybatis
110 | Active | 80 | 4.11.0.1      | Apache ServiceMix :: Bundles :: junit
111 | Active | 80 | 5.5.0.Final_2 | Apache ServiceMix :: Bundles :: drools
…
```

Il peut être nécessaire d'attendre !

Si vous tapez la commande **bundle:list** juste après le démarrage, il se peut que ces bundles soient *en cours de chargement*. Pour le savoir :

- tapez de nouveau la commande.
- si la liste affichée est *plus grande*, c'est que le chargement est en cours.
- si la liste *n'évolue plus*, le chargement est terminé.

Quand le chargement est effectif, vous pouvez examiner la liste des statuts.

Test d'installation "End-to-End" de la DSN

Si vous procédez à l'installation initiale et complète du **Module DSN**, vous avez été redirigé sur ce guide pour procéder à l'installation de l'application "Moteur Déclaratif" : veuillez alors retourner maintenant au *Guide de référence* qui vous a conduit ici, et vous référer au chapitre "**Tester la production du flux via regDSN – Moteur Déclaratif**".

Les instructions complémentaires qui vous sont données là vous permettront de vérifier que *l'ensemble* du module (Produit et Moteur Déclaratif) fonctionne correctement.

Mise à jour d'une version déjà installée

Explications préalables

L'installation d'une nouvelle version (réinstallation) sur une version déjà installée est **différente** d'une installation *initiale* car :

- **Les données BDSN existantes doivent être conservées** et donc potentiellement **remises à niveau** si **la structure de la BDSN a changé.**
- La configuration déjà en place **est conservée**, et n'a donc qu'à être *complétée* pour les éléments nouveaux (s'ils existent).

La procédure à appliquer l'est aussi, et décrite dans ce paragraphe.

Le point d'attention spécifique concerne **les éventuelles modifications structurelles** qu'il vous faut appliquer dans votre changement de version.

Elles concernent l'étape 3 (qui peut donc être optionnelle) de la procédure ci-dessous.

Toutes les règles et tous les principes de prise en charge **des modifications structurelles** sont décrits dans le Chapitre "**Versioning / Versions et Modifications structurelles**" de ce document.

Tout l'historique des changements intervenus au cours des différentes versions y figure aussi.

Ils vous sont rappelés à l'étape 3.

Exemples

Des exemples de transition avec des références d'usage des scripts d'update de la BDSN vous sont donnés dans le chapitre " **Versioning / Versions et Modifications structurelles** ". Lisez-les si c'est votre première mise à jour.

Procédure

La procédure de réinstallation est la suivante :

Etape 1 : Arrêt du Moteur Déclaratif

Etape 2 : Sauvegarde du répertoire d'installation existant

Afin de vous permettre de restaurer l'installation existante en cas d'anomalie, renommez le répertoire regdsn-x.y.z avec le nom de votre choix.

Exemple : regdsn-1.1.10 en regdsn-1.1.10- **backup**

Etape 3 : (Optionnelle) Mise à jour des éléments structurels

Pour procéder :

- à la remise à niveau de la BDSN, vous disposez, dans le répertoire **ddl**, d'**un sous-répertoire update** qui contient les ddls de mise à jour de la BDSN.
- à la valorisation des nouveaux paramètres de configuration éventuels, il suffit de vous référer au paragraphe du chapitre "Configuration" qui les concerne.

Cette étape est dépendante de la transition que vous opérez. Il est possible qu'aucune opération ne soit requise, notamment sur les versions de maintenance.

Les éléments du tableau ci-dessous vous rappellent :

- les règles de passage des scripts d'update
- l'historique des modifications structurelles à examiner pour cette transition.

Pour plus de détail, voir : "**Versioning / Versions et Modifications structurelles**" de ce document.

Pour prendre en compte les modifications structurelles de BDSN et appliquer les scripts d'update associés, il vous faut :

1. Identifier la **version courante du noyau**, **vSource**, repérable sur **le nom du répertoire d'installation** ou en **lançant un shell client** (la version s'affiche sous le logo).

2. Identifier le numéro de **la version du noyau** que vous vous apprêtez **à installer**, **vTarget**, repérable sur **le nom de l'archive tar.gz** livrée.

3. Appliquer en séquence croissante **tous les scripts d'update** pour lesquels **vFrom >= vSource**, et **vTo <= vTarget**.

Lecture	Version du noyau eDSN	Nature du changement	Elément(s) de remise à niveau
^ | | | | | |	4.0	Structure de BDSN - schéma tables *Déclarations*	`edsn.[SGBD].update_3.7.x_4.0.x.sql`
	3.7	Structure de BDSN - schéma tables *Déclarations*	`edsn.[SGBD].update_3.6.x_3.7.x.sql`

(*) Remarque pour les plateformes DB2 et les script "edsn.db2-*.update_3.3.x_3.4.x.sql":

Une erreur s'affichera lors de l'exécution du script concernant une commande 'drop index' sur l'un des 2 indexes

- 'IXdecIdRemuneration' non existant si mise à jour depuis un edsn installé en 3.x

- 'IXdecIdRemuneratio' non existant si mise à jour depuis un edsn installé originellement en 2.x

Veuillez ne pas tenir compte de l'erreur.

Si seule cette erreur est présente, l'exécution du script de mise à jour sera à considérer comme réussie.

Etape 4 : Installation de la nouvelle version livrée

Il convient de transférer et décompresser dans le même répertoire que lors de la précédente installation.

Voir le paragraphe *Mise en place de l'archive : transfert et décompression du fichier de livraison*.

Etape 5 : Redémarrage du Moteur Déclaratif

Etape 6 : Installation des add-ons

Voir le paragraphe "*Installation des add-ons*" de l'installation initiale.

Etape 7 : Destruction de l'installation précédente

Si les étapes précédentes se sont correctement déroulées, vous pouvez procéder à la suppression du répertoire de sauvegarde.

Cas de réinstallation des add-ons

Sur une version déjà installée, il se peut que des add-ons doivent être réinstallés unitairement.

Il *peut arriver* que le nouvel add-on ne soit pas correctement installé : si c'est le cas, il convient de procéder **à un redémarrage en demandant un nettoyage du répertoire de cache** utilisant l'option **clean**.

Soit :

```
bin> ./regdsn clean
```

puis d'arrêter et redémarrer.

Installation sur un autre environnement

Les procédures d'installation (initiale ou réinstallation) sont les **mêmes quel que soit l'environnement concerné**. Il convient d'appliquer exactement les **mêmes instructions**.

Il ne faut pas cloner les installations !

Le **copier/coller** de répertoires Moteur Déclaratif est **strictement prohibé**.

Le Moteur Déclaratif intègre en effet des répertoires techniques de *cache* qui ne fonctionneront plus s'il est dupliqué.

Il ne faut donc pas procéder aux installations secondaires par copie de répertoires.

Configuration

Fichiers de configuration

Les fichiers de configuration du Moteur Déclaratif se trouvent dans le répertoire **regdsn-home/conf**.

L'ensemble de ces fichiers est nécessaire au bon fonctionnement du Moteur Déclaratif, mais **seuls les fichiers décrits ci-dessous sont modifiables**. Ils sont classés par thème

Les autres fichiers de configurations que ceux cités ici ...

Les autres fichiers doivent être laissés intacts.

Rappel préalable sur les fichiers de configuration

Les fichiers de configuration suivent les règles générales suivantes :

> S'ils n'existent pas dans votre répertoire conf/, ils doivent être créés manuellement.

Ces fichiers de configuration sont préservés d'une installation à l'autre : le processus d'installation ne les écrase pas. Ce mécanisme explique que lors de la première installation, **ces fichiers peuvent ne pas exister**. Il conviendra alors de les créer lorsqu'ils seront configurés.

> Leur format de stockage est le format ANSI.

Ils ne doivent pas être enregistrés au format UTF-8, car ils ne seraient alors plus lisibles et engendreraient des anomalies graves de fonctionnement.

Chaque thème de configuration décrit ci-dessous précise le ou les fichiers auquel il s'adresse.

Services JMX et SSH

SSH

Fichier org.apache.karaf.shell.cfg

Paramètres obligatoires

sshPort

Port du service SSH.

Valeur par défaut : 8101

sshHost

Adresse IP de l'interface réseau sur laquelle est ouvert le service SSH.

Indiquez 0.0.0.0 pour ouvrir le service sur toutes les interfaces réseaux.

Paramètre important

sshIdleTimeout

Timeout (en millisecondes) de la session SSH

Valeur par défaut : 1800000

JMX

Fichier org.apache.karaf.management.cfg

Paramètres obligatoires

rmiRegistryPort

Numéro de port du service de registre RMI.

Valeur par défaut : 1099

rmiServerPort

Numéro de port du service JMX.

Valeur par défaut : 44444

serviceUrl

URL d'appel du service JMX. Remplacez "0.0.0.0" par le nom de la machine d'installation pour un accès distant.

Valeur par défaut :

```
serviceUrl =
service:jmx:rmi://0.0.0.0:${rmiServerPort}/jndi/rmi://0.0.0.0:${rmiRegistryPort}/dsn
```

JMX & SSH : usage et utilisateurs

Fichier user.properties

Comptes d'accès

L'accès aux services JMX et SSH nécessite une connexion : les utilisateurs de ces services JMX et SSH sont des **utilisateurs système spécifiques** qui sont déclarés dans le fichier **users.properties.**

Usage

Les usages de ces services sont précisés dans la partie "Exploitation" de ce document.

Connexions JDBC

Comme nous l'avons déjà vu, les connexions JDBC sont paramétrées dans les fichiers suivants :

Fichier com.soprahr.edsn.datasource.cfg pour la connexion à la BDSN.

Fichier com.soprahr.edsn.hraccess.datasource.cfg pour la connexion à la base de données HR Access.

Chaque fichier décrit en fait les propriétés d'un pool de connexions JDBC.

Paramètres obligatoires des fichiers de configuration des connexions JDBC

datasource.jdbc.driver

Nom de la classe driver JDBC permettant la connexion à la base de données.

datasource.jdbc.url

URL JDBC de la base de données.

datasource.user

Nom de l'utilisateur pour se connecter à la base de données.

datasource.password

Mot de passe pour se connecter à la base de données.

datasource.pool.validationQuery

Ordre SQL permettant de valider les connexions du pool.

Validation des pools de connexion

Ce paramètre n'est pas à proprement parler obligatoire pour le fonctionnement. **Cependant**, il est placé dans cette catégorie pour l'explication ci-dessous.

Le Moteur Déclaratif utilise des **pools de connexions JDBC** pour l'accès aux bases de données (base HR Access et BDSN). Lorsque la base de données est *redémarrée*, il se peut que des connexions présentes dans le pool deviennent invalides et engendrent des erreurs lors de leur utilisation.

Pour éviter cela il faut demander au pool de valider les connexions avec un ordre SQL SELECT renvoyant au moins une ligne.

Ce paramètre permet de spécifier l'ordre utilisé.

Il n'existe **pas de valeur par défaut** car l'ordre est *dépendant* de la base utilisée. Les valeurs proposées sont les suivantes :

Pour Oracle ...

```
datasource.pool.validationQuery=SELECT 1 FROM DUAL
```

Pour DB2 ...

```
datasource.pool.validationQuery=SELECT 1 FROM SYSIBM.SYSDUMMY1
```

Pour MS SQL Server

```
datasource.pool.validationQuery=SELECT 1
```

Valorisation de datasource.pool.validationQuery

La configuration de datasource.pool.vaidationQuery est considérée comme **optionnelle**...

Mais cela dépend de la **version du driver JDBC**.

Lorsque le paramètre n'est pas présent, c'est la méthode Connection.isValid() du JDK qui est appelée. Cette méthode est apparue en Java 1.6 : si le driver JDBC est ancien et qu'il a été développé avec un JDK < 1.6, cela peut provoquer une erreur du type :

```
2015-04-30 18:07:23,194 | ERROR | FelixStartLevel | BlueprintContainerImpl
| Unable to start blueprint container for bundle
com.soprahr.edsn.hibernateloader

org.osgi.service.blueprint.container.ComponentDefinitionException: Unable
to initialize bean hibernateStarter

Caused by: java.lang.AbstractMethodError: java/sql/Connection.isValid(I)
```

La méthode la plus sûre consiste donc à renseigner le paramètre avec les valeurs données ci-dessus.

Renfort des mots de passe

Il est possible d'utiliser le paramètre datasource.cryptedPassword à la place de ce paramètre si l'on souhaite chiffrer le mot de passe (voir **Chiffrement des mots de passe**).

Paramètres facultatifs des fichiers de configuration des connexions JDBC

datasource.pool.maxActive

Nombre maximal de connexions actives. Lorsque ce nombre est atteint et qu'une nouvelle connexion est requise, l'application attend qu'une connexion se libère.

Valeur par défaut : **30**

datasource.pool.maxIdle

Nombre maximal de connexions en attente dans le pool.

Valeur par défaut : **15**

Ce paramètre n'est pas utilisé.

datasource.pool.minIdle

Nombre minimal de connexions en attente dans le pool.

Valeur par défaut : **0**

Ce paramètre n'est pas utilisé.

datasource.pool.testOnBorrow

true si les connexions doivent être testées avant d'être empruntées dans le pool, **false** sinon. Ce test n'est fait que si le paramètre datasource.pool.validationQuery est renseigné.

Valeur par défaut : **true**

datasource.pool.testOnReturn

true si les connexions doivent être testées avant d'être remises dans le pool, **false** sinon. Ce test n'est fait que si le paramètre datasource.pool.validationQuery est renseigné.

Valeur par défaut : **false**

datasource.pool.testWhileIdle

true si les connexions doivent être testées lorsqu'elles sont en attente dans le pool, **false** sinon. Ce test n'est fait que si le paramètre datasource.pool.validationQuery est renseigné.

Valeur par défaut : **false**

Paramétrage du connecteur HR Access

Fichier com.soprahr.edsn.hraccess.connector.cfg

extension.tablePrefix

Permet de spécifier, si besoin, le préfixe des tables HR Access accédées par le connecteur.

Valeur par défaut : **HR**

extension.horsPaieDataStructure

Permet de spécifier, si besoin, la structure de données utilisée pour les données "hors paie du mois".

Anomalie d'usage de ZZ

Le nom **ZZ** utilisé dans le standard DSN 3.1 est normalement un nom *réservé Éditeur*.

La configuration ci-dessus permet de gérer les cas où ce nom a été utilisé anormalement en personnalisation.

Valeur par défaut : **ZZ**

Paramétrage de la structure S10

Fichier com.soprahr.edsn.producer.impl.cfg

Ce fichier contient les valeurs des champs communs à tous les flux DSN.

dsn.application.emetteur.sirenEmetteurEnvoi

Siren de l'émetteur de l'envoi S10.G00.01.001

dsn.application.emetteur.nicEmetteurEnvoi

Nic de l'émetteur de l'envoi S10.G00.01.002

dsn.application.emetteur.nomOuRaisonSocialeEmetteur

Nom ou raison sociale de l'émetteur S10.G00.01.003

dsn.application.emetteur.adresse

Numéro, extension, nature et libellé de la voie S10.G00.01.004

dsn.application.emetteur.codePostal

Code postal S10.G00.01.005

dsn.application.emetteur.localite

Localité S10.G00.01.006

dsn.application.emetteur.codePays

Code pays S10.G00.01.007

dsn.application.emetteur.codeDistributionEtranger

Code de distribution à l'étranger S10.G00.01.008

dsn.application.emetteur.complementLocalisationConstruction

Complément de la localisation de la construction S10.G00.01.009

dsn.application.emetteur.serviceDistributionOuComplementLocalisationVoie

Service de distribution, complément de localisation de la voie S10.G00.01.010

dsn.application.contactEmetteur.codeCivilite

Code civilité S10.G00.02.001

dsn.application.contactEmetteur.nomEtPrenomPersonneAContacter

Nom et prénom de la personne à contacter S10.G00.02.002

dsn.application.contactEmetteur.codeDomaineIntervention

Code domaine d'intervention S10.G00.02.003

dsn.application.contactEmetteur.adresseMailContactEmetteur

Adresse mél du contact émetteur S10.G00.02.004

dsn.application.contactEmetteur.adresseTelephonique

Adresse téléphonique S10.G00.02.005

dsn.application.contactEmetteur.adresseFax

Adresse fax S10.G00.02.006

dsn.application.destinataireCRE.sirenEntrepriseDestinataireCompteRenduExploitation

Siren de l'entreprise destinataire du compte rendu d'exploitation S10.G00.03.001

dsn.application.destinataireCRE.nic

Nic de l'établissement destinataire du compte rendu d'exploitation S10.G00.03.002

dsn.application.destinataireCRE.adresseMelDestinataireCompteRenduExploitation

Adresse mél du destinataire du compte rendu d'exploitation S10.G00.03.003

Configuration des logs

Fichier org.ops4j.pax.logging.cfg

Il s'agit d'un fichier au format Log4j.

Exploitation

Scripts du moteur déclaratif

Les scripts permettant de démarrer, arrêter ou interagir avec le moteur déclaratif se trouvent dans le répertoire "bin" du répertoire d'installation.

Avertissement préalable

Certains des *noms de script* figurant ci-dessous peuvent **rentrer en conflit avec des shells ou des alias existant sur votre système**.

Afin de garantir le lancement de ceux présents dans le répertoire bin quand vous y êtes positionnés, nous vous recommandons :

- de spécifier **explicitement le répertoire courant** dans vos lignes de commandes, c'est-à-dire le nom du script précédé de ".\" ou "./"
- d'utiliser le **nom complet du script** (.bat ou .sh).

Exemple : utilisez "**.\start.bat**" au lieu du raccourci "**start**" sous Windows (qui lancerait une nouvelle invite de commande).

Script regdsn / regdsn.bat

Ce script permet de lancer le Moteur Déclaratif en mode **interactif**.

Il est utile lors de l'installation mais, en fonctionnement normal, **on utilisera le script start** qui lance le Moteur Déclaratif en mode démon.

Dans ce mode, on peut piloter le Moteur Déclaratif en lui passant directement des commandes sur la console.

Pour plus de détail sur les commandes, voir le *Langage de commandes*.

Réservé à l'installation !

Pour l'exploitation normale, il convient d'utiliser **le mode *démon*** ci-dessous.

Script start / start.bat

Ce script permet de lancer le Moteur Déclaratif en mode **démon**.

Dans ce cas, **le shell n'est pas directement disponible**.

Pour utiliser des commandes du shell, on passera alors par le script ***client***.

Pour lancer les commandes ...

On utilise le shell **client** décrit ci-dessous.

Script stop / stop.bat

Ce script permet d'arrêter le Moteur Déclaratif quand il a été lancé en mode démon.

Script client / client.bat

Ce script permet de se **connecter** (en SSH) au Moteur Déclaratif et d'avoir ainsi accès aux commandes du shell.

Lorsque l'on lance ce script sans argument, on retrouve alors l'invite de commande, comme lors d'un démarrage avec le script regdsn.

Passer une commande ...

Avec le script Client, il est également possible de **passer en argument une commande**. Dans ce cas, le script exécutera la commande et rendra immédiatement la main.

L'exemple ci-dessous lance la commande addon:list et affiche le résultat :

```
D:\regdsn-1.0.00\bin>client.bat "addon:list"

Logging in as regdsn
Name               | Version                  | Type | Full name
-------------------------------------------------------------------------
hraccess-feature   | 1.0.0                    | kar  | hraccess-feature-1.0.0.kar
oracle-jdbc-driver | 11.1.0.7.0-Production    | jar  | oracle-jdbc-driver-11.1.0.7.0-Production.jar
```

Aide

Pour obtenir l'aide complète sur la commande **client**, utilisez l'option "**--help**".

Une astuce : savoir si le Moteur Déclaratif est prêt...

Les scripts "start" et "stop" du Moteur Déclaratif sont asynchrones et ne fournissent pas de "statut" de l'application (up and running / stopped).

Pour savoir si le Moteur Déclaratif est prêt, on peut utiliser la commande :

```
regdsn.bat status
ou
regdsn.sh status
```

La réponse est 0 si le Moteur Déclaratif est démarré, 1 s'il ne l'est pas.

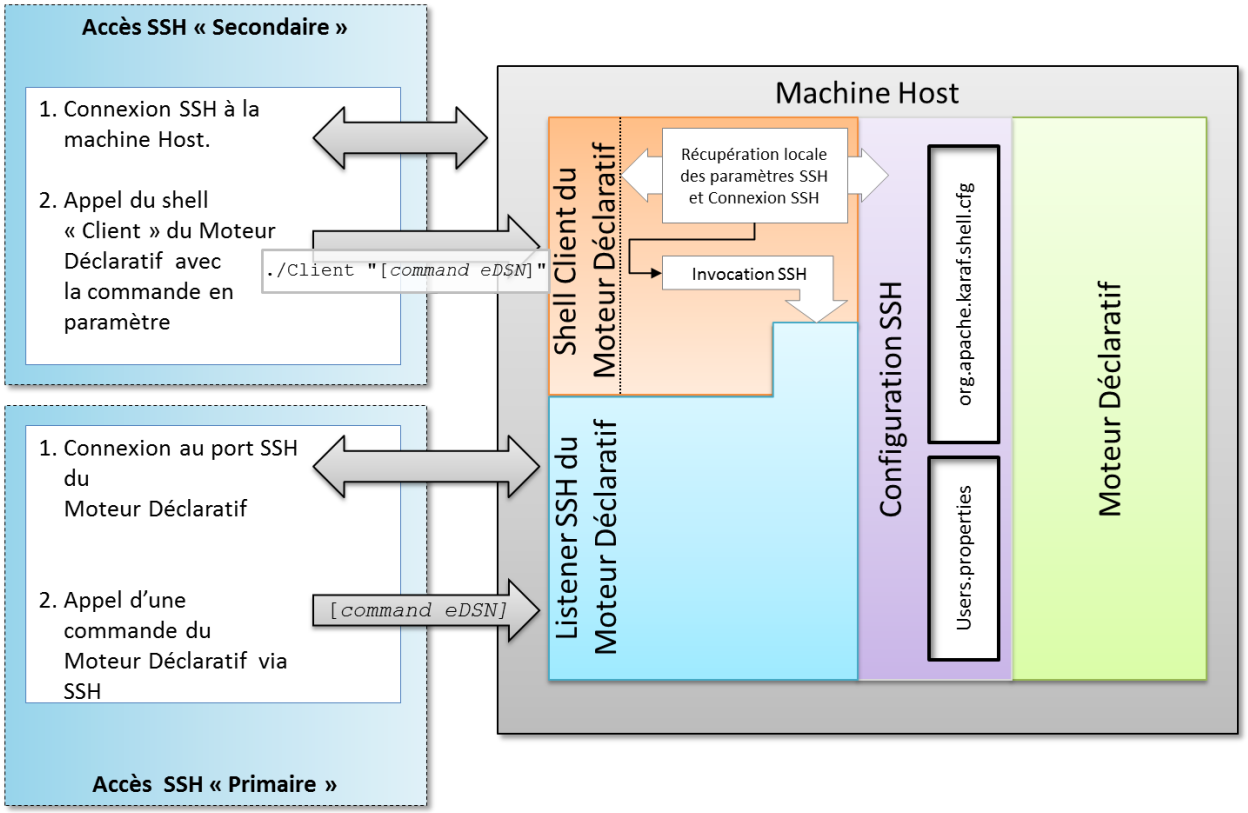
Pilotage du Moteur Déclaratif et SSH

Une fois lancé en mode démon, le Moteur Déclaratif peut être piloté en batch par le **lancement de commandes**.

Ce pilotage utilise le **listener SSH** du Moteur Déclaratif.

Il peut s'effectuer de deux manières différentes.

Schéma général



Accès primaire vs. Accès secondaire

Les accès SSH peuvent être effectués :

- en ***primaire*** sur le port SSH du listener auquel il faut alors se connecter et lancer directement la commande par ce biais.
- en ***secondaire***, via un accès SSH sur la machine hôte, **suivi de l'usage du shell *client*** décrit précédemment. Dans ce cas, c'est l'accès primaire à la machine qui contrôle l'accès.

Dans le cas de l'accès secondaire, le script "**client**" exploite la configuration SSH qui a été spécifiée et établit de manière automatique la connexion au listener SSH sans qu'il soit besoin de se signer.

Usage en production ...

Pour un usage en Production, **l'accès SSH secondaire est requis.**

Langage de Commandes

A propos du langage de commandes

Principes généraux

Le **Moteur Déclaratif** possède son propre langage de commandes.

Ce langage permet de procéder à la production des flux DSN (flux de test ou flux réels), ainsi qu'à des opérations d'exploitation du moteur.

Ce langage puissant permet notamment de s'interfacer avec des outils d'exploitation type ***ordonnanceurs***.

Il est **souple**, pour vous permettre de piloter la plateforme selon vos besoins et vos spécificités d'exploitation.

Usage

Syntaxe

Une commande a une syntaxe de la forme :

```
périmètre: commande [- options] paramètres_obligatoires [ paramètres_optionnels]
```

Avec les principes suivants :

- les éléments entre *crochets* ci-dessus sont **optionnels**
- les **options** doivent toujours figurer **avant les paramètres**
- quand une option **demande une valeur**, celle-ci doit être **précédée d'un espace**

Précautions sur les paramètres

Paramètres commençant par '0'

1. Par précaution, les paramètres **alphanumériques** type *codes* qui ne contiennent **que des chiffres** doivent être "**quotés",** c'est-à-dire encadrés par des quotes (simples ou doubles).

2. Pour les commandes exécutées en mode *client* qui sont déjà écrites en doubles quotes, **les paramètres alphanumériques doivent être mis entre simples quotes**.

Exemples :

- En mode Interactif

dsn:list-ud -s "001" -x -o /TMP/listud.txt 201411

- En mode Démon

./Client "dsn:list-ud -s '001' -x -o /TMP/listud.txt 201411"

En effet, sans cette précaution, les '0' d'entête seront **éliminés**.

Gestion des codes retour

Signification des codes retour

Les commandes renvoient des codes retour qui précisent si elles se sont bien déroulées :

- si la commande s'est effectuée correctement, **le code retour est *égal à* 0**
- si une erreur a été rencontrée pendant l'exécution, le **code retour est *différent* de 0**.

Ces codes retour sont effectifs dans le cas où le Moteur Déclaratif est lancé en **mode démon**, c'est-à-dire dans le cas d'un usage en production.

Corollaire : ils ne sont pas significatifs dans le cas du mode interactif réservé au test.

Lorsque vous écrivez vos scripts d'exploitation, il est **essentiel d'utiliser ces codes retour** afin de **garantir l'enchaînement correct** de vos commandes.

Usage des commandes dans les shells d'exploitation

Ne pas tester les codes retour de commande dans vos shells d'exploitation est **dangereux**...

Référence des codes retour

Les codes retour renvoyés utilisent les plages [**1-49**] et [**100-255**].

Pour vos scripts d'exploitation personnalisés ...

La plage [**50 - 99]** (bornes incluses) n'est pas utilisée et **vous est donc réservée** pour vos scripts personnalisés d'exploitation.

Les codes retour sont décrits dans le tableau ci-dessous (la liste des commandes étant, elle, décrite dans un chapitre ultérieur de ce document).

Ce tableau précise la liste des commandes susceptibles d'émettre le code erreur concerné.

Code	Description	Commandes émettrices
0	Fin normale	Toutes
1	Erreur interne	Toutes
44	Commande non trouvée	N/A
40	Commande invalide (argument, option...)	Toutes
100	Déclaration(s) non trouvée(s)	dsn:dump-messages
100	Déclaration(s) non trouvée(s)	dsn:debug-declaration,dsn:delete-declarations,dsn:delete-matching-declarations
200	Unité(s) déclarative(s) non trouvée(s)	dsn:debug-declaration,dsn:delete-matching-declarations,dsn:gen-file

Commandes non documentées

En utilisant l'auto complétion de l'invite de commandes, vous pourrez trouver des commandes qui **ne sont pas documentées dans ce guide**.

Ces commandes sont réservées à Sopra HR et **ne doivent pas être utilisées**, que ce soit en interactif ou dans des scripts d'exploitation.

Leur fonctionnement n'est en effet ni *pérenne*, ni *garanti*.

Commandes non documentées

Les commandes non documentées dans ce guide **ne doivent pas être utilisées**.

Exploitation Batch

L'exploitation Batch du Moteur Déclaratif consiste à **automatiser des opérations d'exploitation DSN** en utilisant en séquence des appels au langage de commandes de ce moteur.

Ces appels :

- sont soumis au Moteur Déclaratif via son shell Client (voir "*Pilotage du Moteur Déclaratif*").
- s'enchaînent à travers **des scripts** que vous implémenterez en fonction de vos besoins spécifiques d'exploitation.

N'oubliez pas les codes retour

Comme précisé dans le chapitre sur le " *Langage de commandes*", pensez à exploiter les codes retour des commandes dans vos enchaînements afin de garantir le bon déroulement de vos séquences.

Les chapitres qui suivent :

- illustrent des séquences type afin que vous vous familiarisiez avec cette exploitation batch.
- détaillent les commandes dont vous disposez pour écrire vos scripts.

Exemple de séquences d'exploitation

Des séquences à titre d'exemple

Les paragraphes ci-dessous décrivent 2 séquences-type d'usage du moteur déclaratif, sous forme de processus et des commandes de regdsn à utiliser.

Ces séquences sont données à titre d'exemple, pour illustrer les capacités de ces mêmes commandes.

Séquence de production d'une DSN mensuelle

Etape 1 : Calcul du périmètre

Action

Je génère un fichier contenant les identifiants des unités déclaratives de mon périmètre.

Commande(s) de référence

```
dsn:list-ud -x -o FilePath Périmètre PeriodeDSN
```

Création d'un fichier 'FilePath' contenant les identifiants des unités déclaratives correspondant à mon périmètre.

Exemple

```
dsn:list-ud -x -o /tmp/mesUD1409.txt -s ENTPDC 201409
```

Je génère la liste des identifiants de mes unités déclaratives dans le fichier /tmp/mesUD1409.txt pour la société ENTPDC et la période septembre 2014.

Je vais réutiliser mon fichier mesUD1409.txt comme périmètre de génération de mes flux DSN.

Etape 2 : Production du flux

Action

Je produis un flux de prod à partir du périmètre d'unités déclaratives choisi à l'étape 1.

Commande(s) de référence

```
dsn:gen-file -t UDS PREFIXE PROD PerdiodeDSN @FilePath
```

Production d'un flux de production DSN pour le régime NET et un flux DSN pour le régime MSA si des unités déclaratives de mon périmètre correspondent à ces régimes.

Exemple

```
dsn:gen-file -t UDS MONPREFIXE PROD 201409 @/tmp/mesUD1409.txt
```

A partir du fichier /tmp/mesUD1409.txt créé à l'étape 1, je produis en mode PROD mes flux DSN correspondant à ce périmètre. Je vais obtenir un flux DSN pour le régime MSA et/ou un fichier pour le régime NET selon le régime déclaré au sein de mes unités déclaratives de mon périmètre.

Je vais aussi obtenir un fichier compte-rendu correspondant à la liste des déclarations stockées en BDSN et produites dans le ou les flux DSN. Ce compte-rendu est stocké dans regdsn-home/flows.

Les fichiers obtenus sont nommés :

- PREFIXE-201409-dd-mm-aaaa-hh-mm-ss.Pdt.dsn pour les flux de prod.
- PREFIXE-201409-dd-mm-aaaa-hh-mm-ss-compte-rendu.txt pour le compte rendu.

dd-mm-aaaa-hh-mm-ss correspondent à la date et l'heure de production du fichier.

Etape 3 : Contrôle DSN-Val

Action

Je teste au travers de DSN Val les fichiers de produits à l'étape 2 dans le répertoire regdsn-home/flows.

Tant que j'ai des erreurs dans DSN Val, je corrige mes données et reviens à l'étape 2.

Commande(s) de référence

N/A

Exemple

N/A

Etape 4 : Envoi

Action

J'uploade les fichiers .dsn produits à l'étape 2 sur net-entreprise.fr et/ou msa.fr en fonction de leur point de dépôt.

Commande(s) de référence

N/A

Exemple

N/A

Séquence de test d'une DSN mensuelle

Etape 1 : Calcul du périmètre

Action

Je génère un fichier contenant les identifiants des unités déclaratives de mon périmètre.

Commande(s) de référence

```
dsn:list-ud -x -o FilePath Périmètre PeriodeDSN
```

Création d'un fichier 'FilePath' contenant les identifiants des unités déclaratives correspondant à mon périmètre.

Exemple

```
dsn:list-ud -x -o /tmp/mesUD1409.txt -s ENTPDC 201409
```

Je génère la liste des identifiants de mes unités déclaratives dans le fichier /tmp/mesUD1409.txt pour la société ENTPDC et la période septembre 2014.

Je vais réutiliser mon fichier mesUD1409.txt comme périmètre de génération de mes flux DSN.

Etape 2 : Production d'un flux de test

Action

Je produis un flux de test à partir du même périmètre d'unités déclaratives choisi à l'étape 1.

Commande(s) de référence

```
dsn:gen-file -t UDS PREFIXE TEST PerdiodeDSN @FilePath
```

Production d'un flux de production DSN pour le régime NET et un flux DSN pour le régime MSA si des unités déclaratives de mon périmètre correspondent à ces régimes.

Exemple

```
dsn:gen-file -t UDS MONPREFIXE PROD 201409 @/tmp/mesUD1409.txt
```

A partir du fichier /tmp/mesUD1409.txt créé à l'étape 1, je produis en mode PROD mes flux DSN correspondant à ce périmètre. Je vais obtenir un flux DSN pour le régime MSA et/ou un fichier pour le régime NET selon le régime déclaré au sein de mes unités déclaratives de mon périmètre.

Je vais aussi obtenir un fichier compte-rendu correspondant à la liste des déclarations stockées en BDSN et produites dans le ou les flux DSN. Ce compte-rendu est stocké dans regdsn-home/flows.

Les fichiers obtenus sont nommés :

- PREFIXE-201409-dd-mm-aaaa-hh-mm-ss.Pdt.dsn pour les flux de prod.
- PREFIXE-201409-dd-mm-aaaa-hh-mm-ss-compte-rendu.txt pour le compte rendu

dd-mm-aaaa-hh-mm-ss correspondent à la date et l'heure de production du fichier.

Etape 3 : Contrôle DSN-Val

Action

Je teste au travers de DSN Val les fichiers de produits à l'étape 2 dans le répertoire regdsn-home/flows.

Tant que j'ai des erreurs dans DSN Val, je corrige mes données et reviens à l'étape 2.

Commande(s) de référence

N/A

Exemple

N/A

Etape 4 : Envoi de test

Action

J'uploade les fichiers .dsn produits à l'étape 2 sur net-entreprise.fr et/ou msa.fr en fonction de leur point de dépôt.

Commande(s) de référence

N/A

Exemple

N/A

Performances

A l'occasion de la DSN 3.1.30, tout un ensemble d'éléments de consolidation sur les performances et la gestion de la volumétrie ont été intégrés.

Ils ont donné lieu à la création de ce chapitre qui a pour but de regrouper les éléments de description et de préconisations sur ce thème.

A partir de la 3.1.30

L'ensemble des éléments décrits ci-dessous sont valables à partir de la DSN 3.1.30

Accès et Structure BDSN

Colonne periodeDSN

Dans la BDSN, la colonne "**periodeDSN**" a été ajoutée dans la plupart des clés primaires, des index et des clés étrangères.

A titre d'exemple, la table "dsnRemunerationProperty" est modifiée de la manière suivante :

	Avant	**Après**
Clé primaire	(remunerationId, rubriqueCle)	(**periodeDsn**, remunerationId, rubriqueCle)
Index	(declarationId)	(**periodeDsn**, declarationId)
Clé étrangère sur dsnRemuneration	(remunerationId)	(**periodeDsn**, remunerationId)

Accès BDSN et compatibilité au partitionnement

Tous les ordres SQL générés par le Moteur Déclaratif ont été revus de manière à optimiser l'utilisation du critère sur la colonne "periodeDsn".

Cette modification permet une amélioration globale des performances et limitera l'impact de l'augmentation du volume de données dans la BDSN au fil des mois.

Elle permet également de **simplifier le partitionnement des tables de la BDSN sur la colonne "periodeDsn"**.

Nouveaux index

Tables des messages

Sur toutes les tables stockant les messages (information, warning ou erreur) associés aux différents éléments d'une déclaration, un **nouvel index sur (periodeDsn, declarationId)** a été ajouté afin d'en améliorer les temps d'accès.

Table dsnSalarie

Lors de la phase d'extraction d'une déclaration, le Moteur Déclaratif recherche les salariés dans les déclarations du mois précédent en utilisant le matricule des salariés afin de détecter des éventuels changements de situation. Le matricule est stocké dans la colonne functionalId de la table dsnSalarie.

Afin d'améliorer les performances de cette recherche, **un nouvel index sur (periodeDsn, functionalId)** a été ajouté à la table dsnSalarie.

A noter que les index spécifiques que vous auriez pu positionner sur votre BDSN peuvent être supprimés, les nouveaux index proposés assurant un taux de couverture complet.

Mise en œuvre

Comme pour toutes les versions, le Moteur Déclaratif inclut les scripts de mise à jour de la base de données BDSN correspondant à ces modifications.

Il convient cependant de noter qu'étant donné le nombre de modifications effectuées, les scripts de la 3.1.30 peuvent prendre plus de temps que pour les versions précédentes.

Moteur Déclaratif et accès aux données Produit

Les accès formulés par l'Espace DSN aux tables Produit (via son connecteur) ont été optimisés.

Recherche de la situation antérieure au rappel

Le changement opéré sur la valorisation du bloc 41 en cas de rappel a permis de supprimer la recherche de la situation antérieure au rappel, et de ce fait, la requête décrite ci-dessous.

```
select a.DATXXX, a.DATEFF, a.V40002, a.V40003, a.Y40004, a.V40005,
a.V40007, a.V40008, a.V40011, a.N40012, a.N40013, a.V40014, a.M40015,
a.V40016, a.V40017, a.Y40019, a.V40021, a.V40022, a.N40023, a.V40024,
```

```
a.IDMDL6, b.CDNORM
from $$.ZXMJ a
join $$.ZXMI b on a.NUDOSS=b.NUDOSS
join $$.ZX5V c on a.NUDOSS=c.NUDOSS
join $$.ZX00 d on a.NUDOSS=d.NUDOSS
join $$.ZYAD e on d.NUGEST=e.NUDOSS
where b.AAAAMM<:periodeDsn and b.AAAAMM>=:periodeDsnRappele and
b.D40001=:dateDebutContrat and b.X40009=:numeroContrat and
a.DATEFF<=:dateReference and a.DATXXX>=:dateReference and b.FLEXCL<>'X' and
c.VALIDT='D' and e.X30019=:matriculeSalarie
order by b.DT40FN desc, a.NUDOSS desc
```

Cette requête était régulièrement identifiée comme pénalisante dans les traitements de constitution des DSN et pouvait multiplier par 3 les temps constatés.

Autres optimisations

Les autres requêtes sur les tables Produit qui ont été optimisées concernent :

- la sélection de la population

La requête précédemment formulée :

```
select distinct c.NUDOSS as nudossSalarie
from $$.ZXMI a
join $$.ZX5V d on a.NUDOSS = d.NUDOSS
join $$.ZY00 c on a.IDESOC = c.SOCCLE
and a.IDEMAT = c.MATCLE
where a.IDMDL6=:model6 and a.E11001=:codeEtablissement
and a.D05005=:dateDuPremierMoisDePaie
and a.FLEXCL <> 'X'
and d.VALIDT='D' OU and (d.VALIDT='I' or d.VALIDT='X' or d.VALIDT='D')
[and a.CODLOT=:codeLot]
[and a.V00007=:pointDepot]
```

a été remplacée par :

```
select distinct c.NUDOSS as nudossSalarie
from $$.ZXMI a
join $$.ZX5V d on a.NUDOSS = d.NUDOSS and a.PERPAI = d.PERPAI
join $$.ZY00 c on a.IDESOC = c.SOCCLE
and a.IDEMAT = c.MATCLE
```

```
where a.Y11001  = :Siret
and a.AAAAMM = : MoisDePaie
and a.FLEXCL <> 'X'
and d.VALIDT='D' OU and (d.VALIDT='I' or d.VALIDT='X' or d.VALIDT='D')
[and a.CODLOT=:codeLot]
[and a.V00007=:pointDepot]
```

- la récupération des bulletins et contrats

La requête précédemment formulée :

```
select
a.NUDOSS as nudossBulletin, a.CDNORM, a.DATD0J, a.DATF0J, a.IDMDL6,
a.E11001, a.IDESOC, a.IDEMAT, a.FLEXCL, c.NUDOSS as nudossSalarie,
a.D05005, a.D40001, b.DATXXX, b.DATEFF, b.V40002, b.V40003, b.Y40004,
b.V40005, …….. b.IDMDL6 as temoinPerPar, d.VALIDT, g.DTVIRE
from $$.ZXMI a
join $$.ZXMJ b on a.NUDOSS = b.NUDOSS
join $$.ZX5V d on a.NUDOSS = d.NUDOSS
join $$.ZY00 c on a.IDESOC = c.SOCCLE and a.IDEMAT = c.MATCLE
join $$.ZX6B g on a.NUDOSS = g.NUDOSS and g.CODLAN='FR'
where a.IDMDL6=:model6 and a.E11001=:codeEtablissement
and a.D05005=:dateDuPremierMoisDePaie
and a.FLEXCL <> 'X'
and d.VALIDT='D' OU and (d.VALIDT='I' or d.VALIDT='X' or d.VALIDT='D')
[and a.CODLOT=:codeLot]
[and a.V00007=:pointDepot]
and c.NUDOSS in (:nudossSalaries)
order by
nudossSalarie asc,
a.D40001 asc,
a.X40009 asc, b.DATEFF asc,
a.DT40FN asc, nudossBulletin asc
```

a été remplacée par :

```
select
a.NUDOSS as nudossBulletin, a.CDNORM, a.DATD0J, a.DATF0J, a.IDMDL6,
a.E11001, a.IDESOC, a.IDEMAT, a.FLEXCL, c.NUDOSS as nudossSalarie,
a.D05005, a.D40001, b.DATXXX, b.DATEFF, b.V40002, b.V40003, b.Y40004,
b.V40005, …….. b.IDMDL6 as temoinPerPar, d.VALIDT, g.DTVIRE
```

```
from $$.ZXMI a
join $$.ZXMJ b on a.NUDOSS = b.NUDOSS and a.PERPAI = b.PERPAI
join $$.ZX5V d on a.NUDOSS = d.NUDOSS and a.PERPAI = d.PERPAI
join $$.ZY00 c on a.IDESOC = c.SOCCLE and a.IDEMAT = c.MATCLE
join $$.ZX6B g on a.NUDOSS = g.NUDOSS and g.CODLAN='FR' and a.PERPAI =
g.PERPAI
where a.AAAAMM=: MoisDePaie and a.Y11001=:Siret
and a.D05005=:dateDuPremierMoisDePaie
and a.FLEXCL <> 'X'
and d.VALIDT='D' OU and (d.VALIDT='I' or d.VALIDT='X' or d.VALIDT='D')
[and a.CODLOT=:codeLot]
[and a.V00007=:pointDepot]
and c.NUDOSS in (:nudossSalaries)
order by
nudossSalarie asc,
a.D40001 asc,
a.X40009 asc, b.DATEFF asc,
a.DT40FN asc, nudossBulletin asc
```

Référence des commandes

Commandes générales

COMMANDE DSN:LIST-UD

Usage

Cette commande permet d'énumérer la liste des Unités Déclaratives correspondant au périmètre passé en paramètre.

Syntaxe

---

dsn:list-ud

```
[-d PdD] [-l Lot] [-s Entreprise] [-p Pole] [-e Etablissement] [-f Fraction] [-x]
[-o FichierDeRésultat]

  PériodeDSN
```

---

où :

- **PériodeDSN** : Période DSN au format AAAAMM.

- Les options **-d, -l, -s, -p, -e** permettent de spécifier les Unités Déclaratives concernées, en restreignant l'énumération au périmètre spécifié par les critères de filtre correspondants
  - **PdP** : Point de dépôt. Valeurs : NET ou MSA.
  - **Lot** : Code lot de gestion tel que défini dans le système de paie.
  - **Entreprise** : Code entreprise tel que défini dans le système de paie.
  - **Pole** : Code pôle d'établissement tel que défini dans le système de paie.
  - **Etablissement** : Code établissement tel que défini dans le système de paie.
  - **Fraction** : Numéro de fraction des Unités Déclaratives.

- L'option **-o** permet de rediriger le résultat de l'énumération dans le fichier précisé.
  - FichierDeRésultat : nom du fichier dans lequel le résultat doit être produit (utilisez '/' comme séparateur de fichiers, y compris sous Windows).

- L'option **-x** est à utiliser avec l'option -o et permet de spécifier un format de sortie "court" compatible avec le type UDS de la commande dsn:gen-file ("CodeEtablissement/PointDeDepot/LotDeGestion").

Retour

La console affiche la liste des Unités Déclaratives correspondantes :

- Si l'option - x n'est pas spécifiée, la liste est affichée sous la forme d'un tableau contenant : le siret, le point de dépôt, la fraction, le code établissement et le lot de gestion.
- Si l'option - x est spécifiée, on affiche pour chaque Unité Déclarative une ligne au format "CodeEtablissement/PointDeDepot/LotDeGestion".

Si une sortie fichier est demandée, le résultat est redirigé vers ce fichier.

Nouveauté 2.0.60

A partir de la 2.0.60, la liste des éléments renvoyés par la commande en format long (option -x **non** positionnée) **s'enrichit** et permet d'obtenir l'ensemble des éléments descriptifs des Unités Déclaratives de l'exercice DSN concerné. La liste renvoyée est la suivante :

Cela permet de connaître de manière dynamique les périmètres actifs d'un exercice donné (liste des pôles, liste des lots...).

Le retour de la commande est le suivant :

```
Siret | Fraction | Depot | Code etablissement | Etablissement | Code
entreprise | Entreprise | Code pole | Pole  | Code lot | Lot
```

COMMANDE DSN:LIST-DECLARATIONS

Usage

Cette commande permet d'obtenir la liste des DSN stockées en BDSN sur la période et le périmètre spécifiés.

Syntaxe

---

dsn:list-declarations

```
[-m ModeDSN] [-d PdD] [-l Lot] [-s Entreprise] [-p Pole] [-e Etablissement] [-f
Fraction] [-x] [-o FichierDeRésultat]

  PériodeDSN
```

---

où :

- **PériodeDSN** : Période DSN au format AAAAMM.

- Les options -m, -d, -l, -s, -p, -e permettent de spécifier les déclarations concernées, en restreignant l'énumération au périmètre spécifié par les critères de filtre correspondants :
  - **ModeDSN** : TEST, PROD, ou ALL.
    - **A partir de la 2.0.50**, la valeur par défaut est **PROD.**
  - **PdD** : Point de dépôt. Valeurs : NET ou MSA.
  - **Lot** : Code lot de gestion tel que défini dans le système de paie.
  - **Pole** : Code pôle d'établissement tel que défini dans le système de paie.
  - **Etablissement** : Code établissement tel que défini dans le système de paie.
  - **Fraction** : Numéro de fraction des Unités Déclaratives.

- L'option -o permet de rediriger le résultat de l'énumération dans le fichier précisé.
  - **FichierDeRésultat** : Nom du fichier dans lequel le résultat doit être produit (utilisez '/' comme séparateur de fichiers, y compris sous Windows).

- L'option -x permet de spécifier un format de sortie "court" n'affichant que les identifiants techniques des déclarations et compatible avec les commandes prenant en entrée ce type de paramètres.

Retour

La console affiche la liste des déclarations correspondantes :

- Si l'option - x n'est pas spécifiée, la liste est affichée sous la forme d'un tableau contenant : l'identifiant technique de la déclaration, le siret, le point de dépôt, la fraction, le code établissement, le lot de gestion, l'année de déclaration, le mois de déclaration et le statut de la déclaration.
- Si l'option - x est spécifiée, on affiche pour chaque déclaration une ligne contenant l'identifiant technique de la déclaration.

Si une sortie fichier est demandée, le résultat est redirigé vers ce fichier.

COMMANDE DSN:DUMP-MESSAGES

Usage

Cette commande permet de produire un rapport au format csv contenant toutes les informations, anomalies, erreurs associées à la production d'une DSN.

Ces informations, appelées ici **messages**, sont stockées au sein de la DSN et peuvent avoir les origines suivantes :

Connecteur	Le message est émis lors de l'accès aux données sources PRDB (ZX), Réglementaires 'ZD), GA (ZY).
Utilisateur	Le message correspond à la validation d'un utilisateur.
Outil d'autocontrôle	Le message est émis par la brique d'autocontrôle.
Forçage du flux	Le message est émis lorsqu'il y a forçage de flux.

Messages réservés

Les messages Utilisateur, Outil d'autocontrôle, Forçage du flux ne sont jamais émis dans le cas du Moteur Déclaratif.

Syntaxe

dsn:dump-messages

```
[-o RepertoireDeRésultat]
[-m]
<à partir de la DSN 2.0.50> [-n] NomDuFichierRapport
<à partir de la DSN 2.0.50> [-l] Niveau
<à partir de la DSN 2.0.50> [-t] Origine
```

IdDeclarations

où :

- **IdDeclarations** : Liste des identifiants techniques des déclarations à produire qui peut être :
  - Une liste d'identifiants techniques séparés par des espaces.
  - Un fichier contenant la liste des identifiants techniques (un par ligne). Dans ce cas, il s'exprime sous la forme : **@***nom_du_fichier*. Ce fichier peut être obtenu avec la commande dsn:list-declarations.

- L'option **-o** permet de spécifier le nom du répertoire où sera produit le rapport.
  - **RepertoireDeRésultat** : Nom du répertoire dans lequel le ou les fichiers résultats doivent être produits (utilisez '/' comme séparateur, y compris sous Windows).

- L'option **-m** permet de produire un fichier par DSN. Si elle n'est pas spécifiée, tous les messages sont agrégés dans un seul fichier. La nomenclature des fichiers de rapport produits dépend de cette option et est précisée ci-dessous.

- L'option **-n** permet de spécifier le nom du fichier rapport. Elle n'est active que si l'option **-m n'est pas utilisée**.

- L'option **-l** permet de filtrer les messages de niveau supérieur au niveau indiqué. Les niveaux possibles sont, dans l'ordre croissant :
  INFO > **WARNING** > **ERROR.**

- L'option **-t** permet de filtrer les messages sur leur origine. Les origines possibles sont : **CONNECTOR**, **DSN_VAL**, **FORCING**, **USER.**

Options réservées

Dans le cas du Moteur Déclaratif, les valeurs DSN_VAL, FORCING et USER sont sans effet car les messages correspondants ne sont jamais émis.

Retour

La commande produit un ou plusieurs fichiers rapports dans le répertoire **messages**, en fonction de l'utilisation des options - m et -n :

- Si l'option **-m est spécifiée**, la commande produit :
  - un fichier par déclaration de nom : `IdUD.AAAAMM.IdDeclaration-messages.csv` et où IdUD vaut CodeEtablissement.PointDeDepot.LotDeGestion.
- Si l'option **-m n'est pas spécifiée**, la commande produit **un seul fichier** rapport contenant l'ensemble des messages pour toutes les déclarations, dont le nom dépend de l'usage de l'option -n :
  - si l'option -n est spécifiée, elle force le nom du fichier.
  - si l'option -n n'est pas utilisée, le fichier est nommé : horodatageExécution_messages.csv.

Pour ce qui est du fichier des erreurs, la commande produit :

- Si l'option **-m est spécifiée :**
  - un fichier de nom `horodatageExécution-messages-declaration-errors.txt`
- Si l'option **-m n'est pas spécifiée :**
  - si l'option -n est spécifiée, un fichier de nom : `NomDuFichierRapport-messages-declaration-errors.txt`.
  - si l'option -n n'est pas utilisée, un fichier de nom : `HorodatageExécution-messages-declaration-errors.txt`

Le fichier rapport contient un message par ligne qui précise le poids du message, le bloc concerné et un libellé de message, trié par gravité (du poids le plus fort au poids le plus faible).

Chaque ligne est composée des colonnes suivantes :

```
SIRET;Point de dépôt;Fraction;Type de bloc;NIR;Nom;Prenom;Bloc;Bloc n+1;Bloc n+2;Niveau;Type;Origine;Code;Libellé
```

où :

- SIRET;Point de dépôt;Fraction : Identifiants de la déclaration
- Type de bloc : Code du bloc de la norme DSN concerné
- NIR du salarié : NIR du salarié concerné par l'erreur
- Nom du salarié : Nom de famille du salarié concerné par l'erreur
- Prénom du salarié : Prénom du salarié concerné par l'erreur
- Bloc : Description du bloc concerné par l'erreur
- Bloc parent niveau 1 : Description de l'ancêtre de niveau n+1 du bloc concerné par l'erreur (dans la hiérarchie de la norme DSN)
- Bloc parent niveau 2 : Description de l'ancêtre de niveau n+2 du bloc concerné par l'erreur (dans la hiérarchie de la norme DSN)
- Niveau du message : Information, warning ou erreur
- Type de message : Technique ou fonctionnel
- Origine du message : **CONNECTOR**, **DSN_VAL, FORCING, USER**
- Code du message
- Libellé du message

COMMANDE DSN:DETAIL-DECLARATION

Usage

Cette commande permet de détailler une déclaration stockée en BDSN.

Syntaxe

---

dsn:detail-declaration

```
[--ud]

 Perimetre [PériodeDSN]
```

---

où :

- **Perimetre** désigne :
  - un Id de déclaration si l'option --ud n'est pas spécifiée
  - une Unité Déclarative si l'option --ud est spécifiée
- **PériodeDSN** : Période DSN au format AAAAMM.
  - elle est obligatoire si l'option --ud est utilisée.
- L'option --ud est à utiliser pour indiquer que le périmètre est une Unité Déclarative (cf. ci-dessus).

Retour

La console retourne les éléments suivants :

```
Identifiant technique
Nature
Type
Identifiant declaration annulee ou remplacee (si annule et remplace)
Identifiant technique declaration annulee ou remplacee (si annule et
remplace)
Periode
Mode
Norme
Numero d'ordre
Numero fraction
Point depot
```

```
Etablissement
Entreprise
Lot
Pole
Champ declaration
Date d'extraction
```

Commandes liées aux DSN Réelles et aux DSN de Contrôle

COMMANDE DSN:GEN-FILE

Usage

Cette commande permet, en séquence, de réaliser :

- l'extraction des données source de la DSN
- le chargement de ces données dans la BDSN
- la production de fichier DSN correspondant, qui peut être envoyé à la plateforme GIP

Le fichier produit correspond au périmètre spécifié. A noter que :

- le périmètre indiqué dépend du type positionné (Entreprise, Etablissement, Unités Déclaratives, ..) à travers l'option -t
- lorsque que la liste des éléments de ce périmètre est volumineuse, elle peut être spécifiée à travers un fichier (le périmètre commence alors par le caractère @)

Syntaxe

---

dsn:gen-file

```
[-q] [-k] [-t TypeDePérimètre]

RadicalFichiersDSN ModeDSN PeriodeDSN Périmètre
```

---

où

- **RadicalFichierDSN** : Radical de nom qui servira à nommer les différents fichiers produits. 20 caractères maximum.
- **ModeDSN** : TEST ou PROD.
- **PériodeDSN** : Période DSN au format AAAAMM.
- **Périmètre** : Périmètre à traiter (variable selon le type choisi via l'option -t, voir ci-dessous)

- L'option -t permet de spécifier la nature de la liste indiqué dans le paramètre **Périmètre** de la commande :
  - **ETB** : Le périmètre est une liste d'établissements, en codification Produit. C'est la valeur par défaut si l'option n'est pas précisée.
  - **ENT** : Le périmètre est une liste d'entreprises (équivalent fonctionnel du SIREN), en codification Produit.
  - **UDS** : Le périmètre est une liste d'unités déclaratives sous la forme "CodeEtablissement/PointDeDepot/LotDeGestion", en codification **Produit**

pour *CodeEtablissement* et *LotDeGestion*, **NET** ou **MSA** pour le *PointDeDepot*.
    - **POL** : Le périmètre est une liste de pôles d'établissements
    - **LOT** : Le périmètre est une liste de lots de gestion

- Périmètre peut être :
    - Le caractère **'*'** (étoile) : dans ce cas toutes les unités déclaratives connues sont traitées.
    - Une liste de valeurs séparées par des espaces, valeur dépendantes du type spécifié.
    - Un fichier contenant la liste des éléments du périmètre à produire (un élément par ligne). Dans ce cas, il s'exprime sous la forme : **@***nom_du_fichier*

- L'option -k permet de conserver les données déjà présentes en base si elles existent (unités déclaratives déjà traitées sur la même période). Sans cette option les données existantes concernant les mêmes unités déclaratives sur la même période sont supprimées.

- L'option -q permet ne n'afficher aucun message dans la console lors de l'exécution (hormis les messages d'erreurs s'il y en a).

Retour

Les fichiers produits en retour sont :

- RadicalFichierDSN_AAAAMM_Horodatage.NET.dsn : il contient la DSN produite pour le point de dépôt Net Entreprise.
- RadicalFichierDSN_AAAAMM_Horodatage.MSA.dsn : il contient la DSN produite pour le point de dépôt MSA.
- RadicalFichierDSN_AAAAMM_Horodatage-compte-rendu.txt qui contient la liste des unités déclaratives traitées sous la forme "idtechnique statut idfonctionnel",avec :
    - une ud par ligne
    - idtechnique : clef technique de la DSN correspondante.
    - statut qui peut prendre les valeurs :
        - 0 : DSN en erreur
        - 1 : DSN OK
    - idfonctionnel : sous la forme CodeEtablissement/PointDeDepot/LotDeGestion

Les fichiers sont produits dans le sous-répertoire "flows" du répertoire regdsn-home.

COMMANDE DSN:DELETE-DECLARATIONS

Usage

Cette commande permet de supprimer des DSN chargées en BDSN à partir de leurs identifiants techniques.

Syntaxe

---

dsn:delete-declarations

```
[-f]

IdDeclarations
```

---

où

- **IdDeclarations** : Liste des identifiants techniques des déclarations à supprimer qui peut être :
  - Le caractère **'*'** (étoile) : Dans ce cas toutes les déclarations en BDSN seront supprimées.
  - Une liste d'identifiants techniques séparés par des espaces.
  - Un fichier contenant la liste des identifiants techniques (un par ligne). Dans ce cas, il s'exprime sous la forme : **@***nom_du_fichier*

- L'option -f force la suppression sans demander confirmation.

Retour

La console affiche la liste des DSN candidates à la suppression et demande confirmation si l'option -f n'est pas positionnée.

Les DSN sont supprimées si la confirmation est faite.

COMMANDE DSN:DELETE-MATCHING-DECLARATIONS

Usage

Cette commande permet de supprimer des DSN chargées en BDSN et correspondant à un périmètre donné.

Syntaxe

---

dsn:delete-matching-declarations

```
[-f] [-m ModeDSN] [-t TypeDePérimètre]

PériodeDSN Périmètre
```

ù

- **PériodeDSN** : Période DSN au format AAAAMM.
- **Périmètre** : Périmètre à traiter (variable selon le type choisi via l'option -t, voir ci-dessous).

- L'option -t permet de spécifier la nature de la liste indiqué dans le paramètre **Périmètre** :
  - **ETB** : Le périmètre est une liste d'établissements, en codification Produit. C'est la valeur par défaut si l'option n'est pas précisée.
  - **ETR** : Le périmètre est une liste d'entreprises (équivalent fonctionnel du SIREN), en codification Produit.
  - **UDS** : Le périmètre est une liste d'unités déclaratives sous la forme "CodeEtablissement/PointDeDepot/LotDeGestion", en codification Produit pour *CodeEtablissement* et *LotDeGestion*, NET ou MSA pour le *PointDeDepot*.
  - **POL** : Le périmètre est une liste de pôles d'établissements.
  - **LOT** : Le périmètre est une liste de lots de gestion.

- Périmètre peut être :
  - Le caractère **'*'** (étoile) : dans ce cas toutes les unités déclaratives connues sont traitées.
  - Une liste de valeurs séparées par des espaces, valeur dépendantes du type spécifié.
  - Un fichier contenant la liste des éléments du périmètre à produire (un élément par ligne). Dans ce cas, il s'exprime sous la forme : **@***nom_du_fichier*

- L'option -m permet de ne supprimer que les déclarations
  - de test (valeur **TEST**) ou
  - réelle (valeur **PROD**).

- L'option -f force la suppression sans demander confirmation.

Retour

La console affiche la liste des DSN candidates à la suppression et demande confirmation si l'option -f n'est pas positionnée.

Les DSN sont supprimées si la confirmation est faite.

COMMANDE DSN:DEBUG-DECLARATION

Usage

Cette commande permet de debugger une déclaration en produisant un fichier zip contenant les données source de la DSN et son contexte d'exécution.

Syntaxe

---

dsn:debug-declaration

```
[-n] [-q]

IdDeclaration [Salaries]
```

---

où

- **IdDeclaration** : Identifiant de la déclaration à debugger, obtenu après exécution de la commande gen-file (ou via list-declarations pour une DSN déjà produite).

- **Salaries** (optionnel) : Permet de restreindre les données produites à un ou plusieurs salariés de la déclaration en indiquant leurs clés fonctionnelles. Si ce paramètre est omis, tous les salariés de la déclaration sont pris en compte. Salaries peut être :
  - une liste de valeurs séparées par des espaces.
  - un fichier contenant la liste des valeurs (un élément par ligne). Dans ce cas, il s'exprime sous la forme : **@***nom_du_fichier*

- L'option **-n** permet de désactiver l'anynomisation des données. Sans cette option les données suivantes sont anonymisées dans le fichier zip produit : matricule, noms, prénoms, adresse mail.

- L'option **-q** permet de n'afficher aucun message dans la console lors de l'exécution (hormis les messages d'erreurs s'il y en a).

Retour

Le fichier produit en retour est :

- debug_AAAAMM_Horodatage.zip

Le fichier est produit dans le sous-répertoire "debug" du répertoire regdsn-home.

COMMANDE DSN:CANCEL-REPLACE

Usage

Cette commande permet de générer une DSN en mode "Annule et Remplace"

Syntaxe

---

dsn:cancel-replace

```
[-k] [-q]

RadicalFichier ModeDSN OldDeclarationId
```

---

où :

- **RadicalFichierDSN** : Radical de nom qui servira à nommer les différents fichiers produits. 20 caractères maximum.
- **ModeDSN** : TEST ou PROD.
- **OldDeclrationId** : Id de la déclaration à remplacer
- L'option **-k** permet de conserver la précédente déclaration.
- L'option **-q** permet de ne pas afficher les messaqes de progression.

Commandes liées aux Signalements

Explications préalables

Avant de décrire l'ensemble des commandes disponibles, ce paragraphe illustre par l'exemple les principes généraux de fonctionnement des commandes liées aux signalements.

Identifiants des événements de signalements

Les *événements* de signalements sont *codifiés* : leur identifiant fonctionnel résulte de la concaténation des éléments clés qui les constituent.

Il est nécessaire de connaître cet identifiant fonctionnel des événements car il est utilisé en paramètre ou en sortie de certaines commandes.

Changement DSN 3.1 / DSN 3.2

L'identifiant fonctionnel des événements **change entre la DSN 3.1 et 3.2** : la DSN 3.2 privilégie le *NIR* en remplacement du *matricule*.

Mapping

Le matricule pérenne correspond à ZYAD-X30019.

Le NIR correspond à ZYAD-X30001.

Pour les arrêts de travail :

L'identifiant est constitué de :

- L'identifiant du salarié :
  - En version 3.1.3x et inférieure, c'est le ***matricule*** pérenne.
  - En version 3.2 et supérieure, c'est le ***NIR*** s'il existe, le matricule pérenne sinon.
- La **date de début de l'arrêt** (pas la date de dernier jour travaillé, cette date pouvant être identique pour 2 arrêts différents).
- Le témoin de reprise anticipée : **`false`**.

Pour une reprise anticipée :

L'identifiant est constitué de :

- L'identifiant du salarié :
  - En version 3.1.3x et inférieure, c'est le ***matricule*** pérenne
  - En version 3.2 et supérieure, c'est le ***NIR*** s'il existe, le matricule pérenne sinon.
- La ***date de début de l'arrêt*** (pas la date de dernier jour travaillé, cette date pouvant être identique pour 2 arrêts différents).
- Le témoin de reprise anticipée : **`true`**

Pour une fin de contrat :

L'identifiant est constitué de :

- L'identifiant du salarié :
  - En version 3.1.3x et inférieure, c'est le ***matricule*** pérenne.
  - En version 3.2 et supérieure, c'est le ***NIR*** s'il existe, le matricule pérenne sinon.
- La ***date de début de contrat***
- Le ***n° de contrat***

Commandes et conditions de génération de flux

La commande **dsnsig:list-source-events** permet *d'identifier* les événements de gestion qui ont été créés ou modifiés entre deux dates dans HR Access.

La commande **dsnsig:process-events** permet de *traiter* les événements de gestion qui ont été créés ou modifiés entre deux dates et de les prendre en compte dans le Moteur Déclaratif.

Cette commande traite de façon différente les événements selon qu'ils sont traités pour la première fois ou qu'ils ont déjà été traités précédemment.

- Si l'événement est traité pour la première fois, une déclaration de signalement est créée et le flux correspondant est généré.
- Si l'événement a déjà été traité précédemment, une nouvelle déclaration est créée mais le flux n'est pas généré. En effet, le flux à générer dans ce cas dépend de ce que le gestionnaire a fait du précédent flux.
    - S'il n'a pas posté ce flux et qu'il veut en générer un nouveau à partir de la déclaration nouvellement créée, il doit utiliser la commande **dsnsig:gen-file.**
    - S'il a posté ce flux, qu'il n'est pas conforme, et qu'il veut en générer un nouveau à partir de la déclaration nouvellement créée, il doit utiliser la commande **dsnsig:gen-file.**
    - S'il a posté ce flux, qu'il est conforme, et qu'il veut générer un flux **Annule et Remplace** de sa précédente déclaration avec la déclaration nouvellement créée, il doit utiliser la commande **dsnsig:gen-replace-file.**
    - S'il a posté ce flux, qu'il est conforme, et qu'il veut générer un flux **Annule** de sa précédente déclaration avec la déclaration nouvellement créée, il doit utiliser la commande **dsnsig:gen-cancel-file.**

Enchaînements de commandes

Les principes d'enchaînement sont décrits à travers les cas d'usage ci-dessous.

- **Quels sont les événements de gestion de type arrêt de travail saisis ou modifiés entre le 01/07/2015 et le 31/12/2015 inclus ?**

```
regdsn>dsnsig:list-source-events ARRET_TRAVAIL "01072015" "31122015"
```

qui retourne :

Matricule	Début d'absence	Motif de l'arrêt	Dernier jour travaillé	Fin prévisionnelle	Horodatage événement
179089483	03-12-2015	06	02-12-2015	12-12-2015	05-12-2015 19:56:28
179089483	18-08-2015	03	17-08-2015	22-08-2015	21-08-2015 10:44:29
179089483	18-09-2015	01	17-09-2015	20-09-2015	22-09-2015 01:55:20
179089483	20-10-2015	03	19-10-2015	24-10-2015	21-10-2015 21:45:22
179089483	21-11-2015	03	20-11-2015	26-11-2015	25-11-2015 06:10:02

- **Comment créer les déclarations et générer le flux correspondant aux signalements arrêt de travail saisis entre le 01/07/2015 et le 31/12/2015 ?**

Il faut qu'une DSN mensuelle de PROD existe pour les couples (Salarié, contrat).

Ainsi, si on lance la commande ci-dessous sans que ces mensuelles n'existent :

```
regdsn>dsnsig:process-events ARRET_TRAVAIL "01072015" "31122015" PROD
REGDSNAT
```

on obtient :

```
Extraction des donnees...
Aucun contrat trouve dans une déclaration mensuelle pour le salarie
179089483 sur les périodes de 201505 a 201507
Aucun contrat trouve dans une déclaration mensuelle pour le salarie
179089483 sur les périodes de 201507 a 201509
Aucun contrat trouve dans une déclaration mensuelle pour le salarie
179089483 sur les périodes de 201506 a 201508
Aucun contrat trouve dans une déclaration mensuelle pour le salarie
179089483 sur les périodes de 201510 a 201512
Aucun contrat trouve dans une déclaration mensuelle pour le salarie
```

```
179089483 sur les périodes de 201509 a 201511
Aucun contrat trouve dans une déclaration mensuelle pour le salarie
179089483 sur les périodes de 201508 a 201510
Pas de declaration cree
```

- **Comment créer une DSN Mensuelle de PROD incluant les salariés concernés par les arrêts de travail ?**

```
regdsn>dsn:gen-file REGDSN3131 PROD "201507" "ETAB06000_1"
```

qui retourne :

```
Extraction des donnees pour ETAB06000_1/NET/10954582222226_11...
Declaration 676c6178-8ea4-4213-a7c9-526d3eb13f69 cree
Generation du flux...
Fichier D:\DSN\regdsn-home\flows\REGDSN3131-201507-18-02-2016-07-53-
04.NET.dsn cree
Fichier D:\DSN\regdsn-home\flows\REGDSN3131-201507-18-02-2016-07-53-04-
compte-rendu.txt cree Termine.
```

- **Comment créer les déclarations et générer le flux correspondant aux signalements arrêt de travail saisis entre le 01/07/2015 et le 31/12/2015 ?**

La déclaration mensuelle existe désormais, les signalements peuvent être créés.

```
regdsn>dsnsig:process-events ARRET_TRAVAIL "01072015" "31122015" PROD
REGDSNAT
```

retourne :

```
Generation du flux...
Fichier D:\DSN\regdsn-home\flows\REGDSNAT-01072015-31122015-19-02-2016-13-
48-57-NET.dsn cree
Fichier D:\DSN\regdsn-home\flows\REGDSNAT-01072015-31122015-19-02-2016-13-
48-57-event-modified.txt cree
Fichier D:\DSN\regdsn-home\flows\REGDSNAT-summary.txt cree
```

Le fichier REGDSNAT-summary.txt liste les déclarations créées et les identifiants fonctionnels des événements associés :

```
****************************** Declaration(s) creee(s)
********************************
Identifiant fonctionnel : 17908948301072015false ; Identifiant technique :
[e33d32e0-1d81-437c-8328-76b27dc4a989]
Identifiant fonctionnel : 17908948318092015false ; Identifiant technique :
[7181c167-e69a-48c3-81cd-4508371e7471]
Identifiant fonctionnel : 17908948303122015false ; Identifiant technique :
[44a686b0-adc1-4812-8245-6e4fd8e2165a]
Identifiant fonctionnel : 17908948318082015false ; Identifiant technique :
[ddc6a52a-cddc-4b48-80ea-64f6d03c5b07]
Identifiant fonctionnel : 17908948321112015false ; Identifiant technique :
[b48e06ec-bb7d-42cd-ab1b-afa654370ad2]
Identifiant fonctionnel : 17908948320102015false ; Identifiant technique :
[98f3611c-63b8-470a-ba59-1478142c1c4b]
```

- **Quelles sont les déclarations portant sur les arrêts de travail de juillet 2015 ?**

```
regdsn>dsnsig:list-declarations "201507" ARRET_TRAVAIL
```

qui retourne :

Id	Siret	Dépôt	Fraction	Etablissement	Lot	Période	Statut	Mode	Num. Ordre	Id événement
e33d32e0-1d81-437c-8328-76b27dc4a989	109545... |	NET	1/1	ETAB06000_1	10954582222226_11	201507	INITIALIZED	PROD	777	7908948301072015false

- **Comment lister dans le fichier "/tmp/AT201507" les identifiants de déclaration portant sur les arrêts de travail de juillet 2015 ?**

```
regdsn>dsnsig:list-declarations -x -o "/tmp/AT201507" "201507"
ARRET_TRAVAIL
```

Le contenu du fichier est alors :

```
e33d32e0-1d81-437c-8328-76b27dc4a989
```

- **Comment lister les déclarations correspondant à l'événement dont l'identifiant fonctionnel est 17908948301072015false ?**

```
regdsn>dsnsig:list-declaration-by-events "17908948301072015false"
```

qui retourne :

Evénement Id	Déclarations Id
17908948301072015false	e33d32e0-1d81-437c-8328-76b27dc4a989

- **Comment lister dans un fichier "c:/temp/AT201507" les identifiants de déclaration portant sur l'arrêt de travail du 01/07/2015 pour le matricule 179089483 ?**

```
regdsn>dsnsig:list-declaration-by-events -o "c:/temp/AT201507"
"17908948301072015false"
```

Le contenu du fichier est :

```
e33d32e0-1d81-437c-8328-76b27dc4a989
```

Si on repasse une commande de prise en compte d'événements sur la même période de saisie, rien ne se passe car des déclarations existent déjà et les événements de gestion n'ont pas été modifiés.

```
regdsn>dsnsig:process-events ARRET_TRAVAIL "01072015" "31122015" PROD
REGDSNAT
```

```
Extraction des donnees...
Pas de declaration cree
```

- **Comment générer un fichier de flux à partir de cette déclaration ?**

```
regdsn>dsnsig:gen-file REGDSN_FLUX PROD e33d32e0-1d81-437c-8328-
76b27dc4a989
```

- **Comment prendre en compte une modification de l'arrêt de travail du 01/07/2015 faite le 19/02/2016 ?**

```
regdsn>dsnsig:process-events ARRET_TRAVAIL "19022016" "19022016" PROD
REGDSNAT2
```

qui retourne :

```
Fichier D:\DSN\regdsn-home\flows\REGDSNAT2-19022016-19022016-19-02-2016-15-
19-49-event-modified.txt cree
Fichier D:\DSN\regdsn-home\flows\REGDSNAT2-summary.txt cree
```

Il existe désormais deux déclarations concernant l'événement considéré ici.

- **Comment lister les déclarations correspondant à l'événement ?**

```
regdsn>dsnsig:list-declaration-by-events "17908948301072015false"
```

qui retourne :

Evénement Id	Déclarations Id de la plus récemment créée à la plus ancienne
17908948301072015false	*80935b58-7690-49ed-80cc-793bff0b6c1f*;**e33d32e0-1d81-437c-8328-76b27dc4a989**

- **Comment lister les déclarations correspondant à l'événement dans un fichier '/tmp/NEWAT201507' ?**

```
regdsn>dsnsig:list-declarations -x -o "/tmp/NEWAT201507" "201507"
ARRET_TRAVAIL
```

Contenu du fichier :

```
e33d32e0-1d81-437c-8328-76b27dc4a989 80935b58-7690-49ed-80cc-793bff0b6c1f
```

J'avais posté et obtenu la conformité du flux concernant ma première déclaration.

- **Comment générer un flux d'*Annule & Remplace* de mon ancienne déclaration conforme à partir de ma nouvelle déclaration ?**

```
regdsn>dsnsig:gen-replace-file AR_AT PROD "80935b58-7690-49ed-80cc-
793bff0b6c1f;e33d32e0-1d81-437c-8328-76b27dc4a989"
```

qui retourne :

```
Recherche des déclarations...
Mise a jour du mode de la declaration 80935b58-7690-49ed-80cc-
793bff0b6c1f...
Generation du flux...
Fichier D:\DSN\regdsn-home\flows\AR_AT-NET-19-02-2016-15-46-49.dsn cree
```

COMMANDE DSNSIG:LIST-SOURCE-EVENTS

Usage

Cette commande permet de lister l'ensemble des événements de gestion émis à la source pour une période donnée.

Syntaxe

---

dsnsig:list-source-events

typeSignalement dateDebut dateFin

---

où :

- **typeSignalement** : Spécifie le type du signalement à émettre
  - ARRET_TRAVAIL, REPRISE_ARRET_TRAVAIL, FIN_CONTRAT
- **dateDebut :** Spécifie la date de début de période à considérer (date au format JJMMAAAA).
  - Les événements de gestion saisis ou modifiés à partir de cette date incluse sont considérés
- **dateFin :** Spécifie la date de fin e période à considérer (date au format JJMMAAAA).
  - Les événements de gestion saisis ou modifiés jusqu'à cette date incluse sont considérés

Retour

Les événements correspondants sont listés sur la console sous la forme :

```
Matricule | Debut d'absence | Motif de l'arret | Dernier jour travaille |
Fin previsionnelle | Horodatage evenement
-------------------------------------------------------------------------
---------------------------------------
527394334 | 05-12-2014      | 02               | 04-12-2014             |
10-12-2014         | 08-12-2014 15:06:39
```

COMMANDE DSNSIG:PROCESS-EVENTS

Usage

Cette commande permet de traiter les nouveaux événements de signalement disponibles (créés ou modifiés) et de générer les déclarations correspondantes dans la BDSN.

Syntaxe

---

dsnsig:process-events

```
[-q]

 typeSignalement dateDebut dateFin modeDSN radicalFichier
```

---

où :

- L'option **-q** permet de ne pas afficher les messages de progression sur la console.
- **typeSignalement** : Spécifie le type du signalement à émettre
  - ARRET_TRAVAIL, REPRISE_ARRET_TRAVAIL, FIN_CONTRAT
- **dateDebut :** Spécifie la date de début de période à considérer (date au format JJMMAAAA).
  - Les événements de gestion saisis ou modifiés à partir de cette date incluse sont considérés
- **dateFin :** Spécifie la date de fin de période à considérer (date au format JJMMAAAA).
  - Les événements de gestion saisis ou modifiés jusqu'à cette date incluse sont considérés
- **modeDSN :** Permet d'indiquer le mode de DSN à utiliser
  - TEST ou PROD

Retour

Une déclaration est créée pour tous les événements du type de signalement spécifié correspondant à la période comprise entre la date de début et la date de fin spécifiées.

Un flux par point de dépôt est généré, pour tous les événements du type de signalement spécifié correspondant à la période comprise entre la date de début et la date de fin spécifiées et n'ayant pas encore fait l'objet d'une déclaration.

Un fichier summary.txt liste les couples (identifiant fonctionnel d'événement ; identifiant technique de déclaration) pour les déclarations créées.

Un fichier event-modified.txt liste les identifiants fonctionnels des événements ayant déjà été pris en compte.

Remarque

L'exécution de cette commande présuppose l'existence en BDSN d'une déclaration mensuelle du mois précédant la date de l'événement.

COMMANDE DSNSIG:LIST-DECLARATIONS

Usage

Cette commande permet de lister sur critères les déclarations de signalement présentes en BDSN.

Syntaxe

---

dsnsig:list-declarations

```
[-d Dépot] [-e Etablissement] [-l Lot] [-p Pole] [-s Société] [-m Mode] [-o
Fichier_Résultat]
<à partir de la 3.1.10> [-v IdEvenement]

 PeriodeDSN typeSignalement
```

---

où :

- L'option **-d** permet d'indiquer le point de dépôt à considérer
- L'option **-e** permet d'indiquer l'établissement à considérer
- L'option **-l** permet d'indiquer le lot à considérer
- L'option **-p** permet d'indiquer le pôle à considérer
- L'option **-s** permet d'indiquer la société à considérer
- [**3.1.10**] L'option **-v** permet de filtrer les déclarations sur l'Id fonctionnel de l'événement
- L'option **-x** permet d'afficher uniquement les identifiants techniques
- L'option **-m** Permet d'indiquer le mode DSN à considérer
  - ALL, PROD, TEST,
  - Valeur par défaut : PROD.
- L'option **-o** permet de spécifier un fichier de résultat
- **PeriodeDSN** spécifie la période DSN considérée, sous la forme AAAAMM
- **typeSignalement** *Spécifie le type du signalement à émettre*
  - ARRET_TRAVAIL, REPRISE_ARRET_TRAVAIL, FIN_CONTRAT

Retour

Liste les déclarations de signalement correspondant aux critères de sélection, sur la console ou dans un fichier.

COMMANDE DSNSIG:LIST-DECLARATIONS-BY-EVENTS

Usage

Cette commande permet de lister les déclarations de signalement associées à des événements de gestion donnés.

Syntaxe

---

dsnsig:list-declarations-by-events

```
[-o]

 eventIDs
```

---

où :

- L'option **-o** permet de spécifier un fichier de résultat.
- **eventsIDs** désigne une **l**iste d'identifiants fonctionnels d'événements de gestion :
  - soit directement sur la ligne de commande
  - soit dans un fichier préfixé par @

Les eventIDs ont le formalisme suivant :

	Fin de contrat	**Arrêt de travail** **Reprise anticipée**
Format	{MATRICULE}{DATE_DEBUT_CONTRAT}{N°CONTRAT}	{MATRICULE}{DATE_DEBUT_ABSENCE}{TEMOIN_REPRISE_ANTICIPEE}
Exemple	531807946**01112014**900001	527394334**05122014**false

Retour

Liste les déclarations de signalement correspondant aux événements de gestion sur la console ou dans un fichier.

COMMANDE DSNSIG:GEN-FILE

Usage

Cette commande permet de générer des flux de signalement de type "Initial" à partir d'une liste de déclarations présentes en BDSN.

Syntaxe

---

dsnsig:gen-file

```
[-q] [-m]

 typeSignalement dateDebut dateFin
```

---

où :

- L'option **-q** permet de ne pas afficher les messages de progression sur la console.
- L'option **-m** permet de produire un fichier par déclaration.
  - Valeur par défaut : **false**.
- **radicalFichier** permet d'indiquer le radical des fichiers générés par l'exécution de la commande.
- **modeDSN** permet d'indiquer le mode de DSN à utiliser :
  - TEST ou PROD
- **declarationIDs** désigne la liste des identifiants des déclarations pour lesquelles un flux doit être généré :
  - soit directement sur la ligne de commande
  - soir séparés par des ;
  - soit dans un fichier préfixé par @

Exemples :

```
dsnsig:gen-file FIC_TEST TEST ID_declaration1;ID_declaration2
dsnsig:gen-file FIC_TEST TEST @nom_de_fichier
```

Retour

Un flux par point de dépôt est généré pour toutes les déclarations listées.

COMMANDE DSNSIG:GEN-REPLACE-FILE

Usage

Cette commande permet de générer des flux de signalement de type "Annule et Remplace" à partir d'une liste de déclarations présentes en BDSN.

Cette commande ne s'applique que pour les couples de signalements **portant sur le même événement de gestion**.

Syntaxe

---

dsnsig:gen-replace-file

```
[-q]

radicalFichier modeDSN declarationIDsCouple
```

---

où :

- L'option **-q** permet de ne pas afficher les messages de progression sur la console.
- **radicalFichier** permet d'indiquer le radical des fichiers générés par l'exécution de la commande.
- **modeDSN** permet d'indiquer le mode de DSN à utiliser :
  - TEST ou PROD
- **declarationIDsCouple** désigne une liste de couples d'identifiants de déclarations pour lesquelles un flux A&R doit être généré :
  - soit directement sur la ligne de commande
  - soit dans un fichier préfixé par @

declarationIDsCouple a le formalisme suivant :

```
IDNew;IDOld
```

Retour

Les déclarations d'identifiant IDNew sont topées **ANNULE_ET_REMPLACE.**

Un flux par point de dépôt est généré pour toutes ces déclarations.

COMMANDE DSNSIG:GEN-CANCEL-FILE

Usage

Cette commande permet de générer des flux de signalement de type "Annulation" pour les signalements de fin de contrat.

Syntaxe

---

dsnsig:gen-cancel-file

```
[-q]

 radicalFichier modeDSN declarationIDs
```

---

où :

- L'option **-q** permet de ne pas afficher les messages de progression sur la console.
- **radicalFichier** permet d'indiquer le radical des fichiers générés par l'exécution de la commande.
- **modeDSN** permet d'indiquer le mode de DSN à utiliser :
  - TEST ou PROD
- **declarationIDs** désigne une liste de couples d'identifiants de déclarations à annuler :
  - soit directement sur la ligne de commande
  - soit dans un fichier, préfixé par @

Exemples :

```
dsnsig:gen-cancel-file FIC_TEST TEST
ID_declaration_Annule_1;ID_declaration_Annulée_1
ID_declaration_Annule_1;ID_declaration_Annulée_2
dsnsig:gen-cancel-file FIC_TEST TEST @nom_de_fichier
```

Retour

Un flux par point de dépôt est généré pour toutes les déclarations listées.

COMMANDE DSNSIG:DELETE-DECLARATIONS

Disponibilité 3.1.10

Cette commande a été introduite en DSN 3.1.10.

Usage

Cette commande permet de supprimer des DSN de signalement chargées en BDSN à partir de leurs identifiants techniques.

Syntaxe

---

dsn:delete-declarations

```
[-f]

  IdDeclarations
```

---

où :

- **IdDeclarations** : Liste des identifiants techniques des déclarations à supprimer qui peut être :
  - Le caractère **'*'** (étoile) : dans ce cas, toutes les déclarations en BDSN seront supprimées
  - Une liste d'identifiants techniques séparés par des espaces
  - Un fichier contenant la liste des identifiants techniques (un par ligne). Dans ce cas, il s'exprime sous la forme : **@***nom_du_fichier*

- L'option -f force la suppression sans demander confirmation.

Retour

La console affiche la liste des DSN candidates à la suppression et demande confirmation si l'option -f n'est pas positionnée.

Les DSN sont supprimées si la confirmation est faite.

COMMANDE DSNSIG:DELETE-MATCHING-DECLARATIONS

Usage

Cette commande permet de supprimer des DSN de signalement chargées en BDSN et correspondant à un périmètre donné.

Disponibilité 3.1.10

Cette commande a été introduite en DSN 3.1.10.

Syntaxe

dsn:delete-matching-declarations

```
[-f] [-m ModeDSN] [-t TypeDePérimètre]

  PériodeDSN TypeSignalement Périmètre
```

où :

- **PériodeDSN** : Période DSN au format AAAAMM.
- **TypeSignalement** : Type de signalement
  - ARRET_TRAVAIL, REPRISE_ARRET_TRAVAIL, FIN_CONTRAT
- **Périmètre** : Périmètre à traiter (variable selon le type choisi via l'option -t, voir ci-dessous).

- L'option -t permet de spécifier la nature de la liste indiquée dans le paramètre **Périmètre** :
  - **ETB** : Le périmètre est une liste d'établissements, en codification Produit. C'est la valeur par défaut si l'option n'est pas précisée.
  - **ETR** : Le périmètre est une liste d'entreprises (équivalent fonctionnel du SIREN), en codification Produit.
  - **UDS** : Le périmètre est une liste d'unités déclaratives sous la forme "CodeEtablissement/PointDeDepot/LotDeGestion", en codification Produit pour *CodeEtablissement* et *LotDeGestion*, NET ou MSA pour le *PointDeDepot*.
  - **POL** : Le périmètre est une liste de pôles d'établissements.
  - **LOT** : Le périmètre est une liste de lots de gestion.

- Périmètre peut être :
  - Le caractère **'*'** (étoile) : dans ce cas, toutes les unités déclaratives connues sont traitées
  - Une liste de valeurs séparées par des espaces, valeurs dépendantes du type spécifié
  - Un fichier contenant la liste des éléments du périmètre à produire (un élément par ligne). Dans ce cas, il s'exprime sous la forme : **@***nom_du_fichier*

- L'option -m permet de ne supprimer que les déclarations :
  - de test (valeur **TEST**) ou
  - réelles (valeur **PROD**).

- L'option -f force la suppression sans demander confirmation.

Retour

La console affiche la liste des DSN candidates à la suppression et demande confirmation si l'option -f n'est pas positionnée.

Les DSN sont supprimées si la confirmation est faite.

```
Extraction des données...
Aucun contrat trouve dans une declaration mensuelle pour le salarie
179089483 sur les periodes de 201505 a 201507
Aucun contrat trouve dans une declaration mensuelle pour le salarie
179089483 sur les periodes de 201507 a 201509
Aucun contrat trouve dans une declaration mensuelle pour le salarie
179089483 sur les periodes de 201506 a 201508
Aucun contrat trouve dans une declaration mensuelle pour le salarie
179089483 sur les periodes de 201510 a 201512
Aucun contrat trouve dans une declaration mensuelle pour le salarie
179089483 sur les periodes de 201509 a 201511
Aucun contrat trouve dans une declaration mensuelle pour le salarie
179089483 sur les periodes de 201508 a 201510
Pas de declaration cree
```

Autres commandes

COMMANDE SQLSTAT:START-REGISTER

Usage

Cette commande active l'enregistreur d'ordres SQL qui permet d'analyser l'ensemble des ordres SQL exécutés par l'application tout le temps où il est actif. Ces statistiques sont produites sous forme d'un fichier Excel quand l'enregistreur est stoppé via la commande correspondante **stop-register.**

Syntaxe

---

sqlstat:start-register

Options et arguments

Aucun.

Retour

N/A

COMMANDE SQLSTAT:STOP-REGISTER

Usage

Cette commande désactive l'enregistreur d'ordres SQL s'il a été activé, et produit le fichier Excel des statistiques d'accès SQL de l'application.

Syntaxe

sqlstat:stop-register

Options et arguments

Aucun.

Retour

Cela produira un fichier de statistiques de nom :

sqlstats-[*timestamp*].xlsx

dans le répertoire :

$edsn-home/sqlstats

Annexes

Chiffrement des mots de passe

Principe

Sur les environnements de production, il est généralement proscrit de faire apparaître des mots de passe en clair dans les fichiers de configuration.

C'est le cas par exemple du paramètre datasource.password de la connexion JDBC. Dans ce cas, on peut remplacer le paramètre "datasource.password" par le paramètre "datasource.cryptedPassword" contenant une version chiffrée du mot de passe. Les étapes suivantes décrivent la manière de l'obtenir.

Il existe d'autres paramètres type mots de passe qui possèdent leur équivalent encrypté. Ils sont alors précisés dans les chapitres de configuration concernés.

Méthodologie

Avant de chiffrer les mots de passe, on commence par renseigner le paramètre "en clair" (ex : "datasource.password" pour le cas de la connexion JDBC) dans les fichiers de configuration concernés.

Après le démarrage de l'application DSN, on s'assure que tout fonctionne correctement.

Puis on utilise la commande interactive "encryption:encrypt-password" depuis le shell de l'application pour chiffrer chacun des mots de passe :

```
dsn>encryption:encrypt-password Password: t+zOLpMEeyoBTGMV+Rh8gw==
```

Une fois le mot de passe entré, la version chiffrée s'affiche. Il suffit alors de la copier, puis de remplacer dans le fichier adéquat le paramètre en clair par le paramètre en version cryptée (Ex : "datasource.cryptedPassword") avec la valeur copiée.

Par exemple :

```
datasource.jdbc.driver=oracle.jdbc.driver.OracleDriver
datasource.jdbc.url=jdbc:oracle:thin:@ukx00684.ptx.fr.sopra:1521:DSN01
datasource.user=toto
datasource.cryptedPassword=t+zOLpMEeyoBTGMV+Rh8gw==
```

Lorsque tous les mots de passe ont été chiffrés, il est nécessaire de redémarrer l'application.