Import multi-destination

2023-12-19

Besoin

- Pouvoir importer des données vers plusieurs destinations :
  - **Synthèse** (Observation)
  - **Occhab** (Station / Habitat)
  - **Occtax** (Relevé / Occurence / Dénombrement)
  - **Meta-données** (Cadre d'acquisition / Jeu de données)
  - **Monitoring** (Groupe de site / Site / Visite / Observation)
  - autres... (besoin d'évolutivité)
- Pouvoir importer vers plusieurs entités (*e.g.* Station / Habitat)

Choix principaux retenus

Module

- Évolution du module d'import actuel (import vers la synthèse)
- Intégration du module d'import au cœur de GeoNature

Fichiers

- Un seul fichier contenant plusieurs entités
  (*e.g.* Station et Habitats, y compris sur la même ligne)
- Il reste possible d'effectuer deux imports
  (*e.g.* un fichier avec les Stations, un fichier avec les Habitats)

Permissions

- Accès au module : `CRUD IMPORT/IMPORT`
- Gestion des modèles : `CRUD IMPORT/MAPPING`
- Droit d'import dans les JDD : `C` du module de destination

Fonctionnement actuel du module d'import

- Liste des imports
- Processus d'import
  - Modal de sélection du JDD
  - Téléversement du fichier
  - Configuration du fichier (séparateur, encodage, SRID, ...)
  - Correspondance des champs
  - Correspondance des nomenclatures
  - Prévisualisation avant import (bounding box, données)
- Rapport d'import

Évolution de l'interface

Liste des imports

- Liste commune à toutes les destinations (sans informations spécifiques)
- Possibilité de filtrer par destination

Rapport d'import

- Affichage des statistiques spécifiques à chaque destination
- Affichage de composant spécifique (*e.g.* répartition des taxons)

Évolution du processus d'import

Modal de création d'un import

- Sélection de la destination uniquement

Téléversement du fichier

- Choix du jeu de données, en fonction de la destination

Configuration du fichier

- Potentiels paramètres spécifiques

Correspondance des champs

- Regroupement des champs par entités dans des onglets

Prévisualisation avant import

- Tableau de prévisualisation par entités dans des onglets

Fonctionnement actuel du module

Structure de la base de données :

- `bib_fields`, `bib_themes`
- `bib_errors_type`, `t_user_errors`
- `t_mappings`
- `t_imports`
- `t_imports_synthese` (table transitoire)

Processus d'import :

- Téléversement du fichier
- Chargement dans la table transitoire à partir des correspondance de champs
- Contrôle des données et marquage des lignes erronées (asynchrone)
- Prévisualisation des données
- Import des données dans la table de destination (asynchrone)

Évolutions de la base de données

Nouvelles tables

- `bib_destinations`, référencée depuis :
  - `t_imports`
  - `t_mappings`
  - `bib_fields`
- `bib_entities`
- `cor_entity_field` : Un champs peut être commun à plusieurs entités (*e.g.* `id_station`)

Table `t_imports`

- Suppression de la colonne `taxa_count`
- Ajout d'une colonne JSON `statistics`

Table `bib_fields`

- Unicité du coupe (`id_destination`, `name_field`)

Nouvelle destination Occhab

- Ajout d'Occhab à `bib_destinations`
- Ajout de Station et Habitat à `bib_entities`
- Déclaration des champs d'Occhab dans `bib_fields`
- Association des champs aux entitiés dans `cor_entity_field`
- Ajout de la table transitoire `t_import_occhab`, colonnes :
  - `id_import`
  - `line_no`
  - `station_valid`
  - `habitat_valid`
  - champs...

Interface import / destination

- Module polymorphique
- 3 encres : check, import, remove

```
class SyntheseModule(TModules):
    __mapper_args__ = {
        "polymorphic_identity": "synthese",
    }

    _imports_ = {
        "check_transient_data": check_data,
        "import_data_to_destination": import_data,
        "remove_data_from_destination": remove_data,
    }
```

Évolutions techniques

- URL de l'API préfixés par le code de la destination
- Généricité des contrôles : la destination précise sur quel champs le contrôle doit être effectué
- Généricité des notifications :
  - Templates génériques communs
  - Ajout de la destination au contexte Jinja
  - Possibilité de les personaliser par destination

Gestion des entités

- Les colonnes de validité (*e.g.* `station_valid`) sont des colonnes booléennes nullables :
  - `NULL` : la ligne ne contient pas l'entité
  - `False` : la ligne contient l'entité mais elle contient des erreurs
  - `True` : la ligne contient l'entité et celle-ci peut être importé
- Une ligne contient l'entité dès lors qu'une des colonnes de l'entité n'est pas `NULL`.
- Nouvelle erreur `ORPHAN_ROW` : ligne sans entité identifiée
- Les entités sont rattachées entre elles de différente manière :
  - Par `UUID`
  - Par « `entity_source_pk` » : identifiant chez la source
    Exemple : nom de la station pour Occhab
  - Sur la même ligne
    Exemple : import dans Occtax d'un export synthèse
- Les entités sont ordonnées ; on importe d'abord les entités parentes (*e.g.* les stations avant les habitats)

Autres évolutions

JDD ligne par ligne

- Une colonne peut contenir l'UUID du JDD auquel rattacher la ou les entité(s)
- Les permissions sont contrôlées ligne par ligne
- Nouvelles erreurs : `DATASET_NOT_FOUND` et `DATASET_NOT_AUTHORIZED`
- Le JDD global devient optionnel, et prévaux lorsque renseigné

Colonnes virtuelles constantes (#500)

- Lors du mapping des champs, possibilité de spécifier une valeur constante au lieu d'une colonne
- Exemple d'utilisation :
  - Ajout de nomenclature non présente dans le fichier mais commune à celui-ci (méthode d'observation, . . .)
  - Ajout de la colonne `cd_nom` pour des fichiers contenant une seule espèce
  - Spécification du JDD