Data4citizen

Documentation API

Auteur : Frédéric Grenier

Date : 06/06/2017

Document : Data4citizen-Documentation-API.docx

BPM-Conseil

Contents

1 Introduction

Les API de Data4citizen permettent d'exposer la plupart des fonctionnalités de la plateforme aux clients API. Les API permettent d'intéragir avec les éléments visibles depuis l'interface du site (jeux de données, groupes, tags) ainsi qu'avec l'infrascture de stockage (FileStore et DataStore).

2 Présentation générale

L'accès aux API s'effectuent via une URI normalisée possédant la structure suivante :

```
http://[serveur-data4citizen][:port]/api/3/
```

Les appels API se font sous la forme d'un appel HTTP POST en joignant une liste optionnelle de paramètres dans un dictionnaire JSON.

Data4citizen répond aux appels API sous la forme d'un dictionnaire de valeurs JSON.

Par exemple, l'appel suivant via Curl :

```
curl http://[serveur-data4citizen]/api/3/action/group_list
```

Permet d'obtenir une réponse de Data4citizen au format suivant :

```
{
    "help": "...",
    "success": true,
    "result": [
        "data-visualisation",
        "transports",
        "economie "
    ]
}
```

La réponse est un dictionnaire JSON contenant 3 clés :

1. `"help": "..."` : l'url permettant d'accéder à la documentation de la fonction API appelée
2. `"success": true` or `false` : l'API doit normalement toujours retourner un code HTTP *« 200 OK »*. Les erreurs d'exécution sont indiquées par une valeur *« false »* de la clé *« success ».*
3. `"result"` : cette clé contient le résultat de la fonction appelée. Dans le cas de l'exemple d'un appel à la fonction group_list ci-haut, la clé *result* contient une liste de chaînes de caractères reprenant les noms systèmes des groupes.

Les actions API en lecture peuvent être effectuées par toute personne disposant de l'URL. Dans ce cas, seuls les contenus « publics » seront affichés.

Afin de pouvoir insérer des données dans Data4citizen grâce à l'API, il faut disposer de la clé API d'un utilisateur possédant les droits nécessaires.

3 Liste des fonctions (ou actions) de l'API :

Le tableau suivant présente une synthèse des principales actions de l'API standard.

Action	Description
help_show ?name=…	Affichage de la page d'aide d'une fonction API
group_list	Affichage de la liste des groupes Data4citizen
tag_list	Affichage de la liste des tags
license_list	Affichage de la liste des licences Open Data
package_list	Affichage de la liste des noms systèmes des jeux de données.
package_search	Recherche dans Data4citizen. Utilise la syntaxe du moteur Solr utilisé dans la plateforme
package_show	Affichage des métadonnées et des fichiers associées à un jeu de donnée (par nom ou id)
package_create	Création d'un nouveau jeu de données
current_package_list_with_resources	Affiche la liste des jeux de données et de leurs fichiers. Triée par date de modification (les plus récents en premier)
resource_search	Recherche d'un fichier de données
resource_show	Affichage des métadonnées d'un fichier de données
resource_create	Ajoute un fichier de données à un jeu de données
resource_update	Met à jour un fichier de données existant

3.1 help_show

Affiche la page d'aide d'une fonction API.

Paramètre : name – nom de la fonction à afficher

Exemple :

`http://[srv-data4citizen]/api/3/action/help_show?name=package_list`

```
{
        "help": "http://[srv-data4citizen]/api/3/action/help_show?name=help_show",
        "success": true,
        "result": "Return a list of the names of the site's datasets (packages).
                :param limit: if given, the list of datasets will be broken into
                        pages ofat most ``limit`` datasets per page and only one
                        page will be returned at a time (optional)
                :type limit: int
                :param offset: when ``limit`` is given, the offset to start
                      returning packages from
                :type offset: int
                :rtype: list of strings"
}
```

3.2 group_list

Affichage de la liste des groupes définis dans Data4citizen.

Paramètres :

sort (texte) – tri (valeur par défaut : « name asc »; valeurs possibles 'name' ou 'package_count', 'title')

limit (numerique) – nombre de groupes par page de résultat

offset (numerique) – complément de limit pour l'affichage des pages de résultat suivantes

all_fields (booléen) – affichage de la liste des noms de groupe ou du dictionnaire de données complet (valeur par défaut : false)

Exemple :

`http://[srv-data4citizen]/api/3/action/group_list?all_fields=true`


```
{
        "help": " http://[srv-data4citizen]/api/3/action/help_show?name=group_list",
        "success": true,
        "result": [{
                "display_name": "Transports",
                "description": "Données sur les transports",
                "image_display_url": "",
                "package_count": 3,
                "created": "2017-03-16T10:03:53.873303",
                "name": "transports",
                "is_organization": false,
                "state": "active",
                "image_url": "",
                "type": "group",
                "title": "Transports",
                "revision_id": "39ea8aaf-a473-467c-be30-303cb906d473",
                "num_followers": 0,
                "id": "7b0d1853-cf15-4a7d-94ce-1b9544859ada",
                "approval_status": "approved"
        }]
}
```

3.3 tag_list

Affichage de la liste des « étiquettes » utilisées dans Data4citizen. Par défaut, cette fonction renvoie uniquement la liste des étiquttes ne faisant pas partie d'un vocubulaire.

Les étiquettes non liées à un vocabulaire sont les termes saisis dans la zone « Mots-clés » des métadonnées des jeux de données.

Paramètres :

query (texte) – recherche sur le nom de l'étiquette (exemple : ?query=éco)

vocabulary_id (texte) – id ou nom d'un vocabulaire

all_fields (booléen) – affichage de la liste des noms d'étiquettes ou du dictionnaire de données complet (valeur par défaut : false)

Exemple :

`http://[srv-data4citizen]/api/3/action/tag_list`


```
{
"help": "http://[srv-data4citizen]/api/3/action/help_show?name=tag_list",
"success": true,
"result": ["développement", "économie", "métropole"]
}
```

3.4 license_list

Affichage de la liste des licences utilisées dans Data4citizen.

Paramètres : **Aucun**

Exemple :

`http://[srv-data4citizen]/api/3/action/license_list`


```
{
        "help": "http://[srv-data4citizen]/api/3/action/help_show?name=license_list",
        "success": true,
        "result": [{
                "status": "active",
                "maintainer": "Etalab",
                "od_conformance": "not reviewed",
                "family": "",
                "osd_conformance": "not reviewed",
                "domain_data": false,
                "title": "Licence Ouverte / Open Licence version 2.0",
                "url": "http://www.etalab.gouv.fr/licence-ouverte-open-licence",
                "is_generic": true,
                "is_okd_compliant": false,
                "is_osi_compliant": false,
                "domain_content": false,
                "domain_software": false,
                "id": "fr-lo-2.0"
        },
        {
                "status": "active",
                "maintainer": "Etalab",
                "od_conformance": "not reviewed",
                "family": "",
                "osd_conformance": "not reviewed",
                "domain_data": false,
                "title": "Licence Ouverte / Open Licence version 1.0",
                "url": "http://www.etalab.gouv.fr/licence-ouverte-open-licence",
                "is_generic": true,
                "is_okd_compliant": false,
                "is_osi_compliant": false,
                "domain_content": false,
                "domain_software": false,
                "id": "fr-lo-1.0"
        }
}
```

3.5 package_list

Affichage de la liste des noms des jeux de données (packages).

Paramètres :

limit (numérique) – nombre de résultats renvoyés

offset (numérique) – point de départ de la pagination (/!\ : attention, les N premiers dataset sont numérotés de 0 à N-1, pour obtenir les suivants, offset doit être égal à N)

Exemple :

`http://[srv-data4citizen]/api/3/action/package_list?limit=5`

`http://[srv-data4citizen]/api/3/action/package_list?limit=5&offset=5`


```
{
  "help": "http://[srv-data4citizen]/api/3/action/help_show?name=package_list",
  "success": true,
  "result": [
    "dataset-ventes",
    "donnees-gare-voyageurs",
    "kpi_test",
    "mon-dataset-avec-espace-et-majuscule",
    "mon-nouveau-dataset-avec-espace"
  ]
}
```

3.6 package_search

Recherche les jeux de données qui satisfaient le(s) critère(s) de recherche fourni(s). La recherche renvoie un dictionnaire constitué des métadonnées des jeux de données (*packages*) et de leurs fichiers de données (*resources*)

Cette action utilise le moteur de recherche Solr qui est intégré dans Data4citizen. Il faut donc utiliser la syntaxe Solr. Le lien suivant présente les principaux paramètres acceptés par le moteur : https://wiki.apache.org/solr/CommonQueryParameters

Paramètres :

q (texte) – la requete solr sous la forme champ:valeur. Valeur par défaut : *:* (exemple : q=title:kpi)

fq (texte) – filtre de requête (optionnel).

sort (texte) – tri des résultats de recherche. (optionnel) Valeur par défaut : 'relevance asc, metadata_modified desc'. Comme indiqué par la documentation Solr, il s'agit d'une liste de champs et de leur ordre de tri, séparés par des virgules

rows (numérique) – nombre de résultats renvoyés (optionnel) Valeur par défaut : 10

start (numérique) – point de départ de la pagination. (optionnel) Valeur par défaut : 0

facet (texte) – activation des facettes (optionnel). Valeur par défaut : True

facet.field (texte) – liste des champs pour les facettes (optionnel). Exemple : facet.field=["license_title","state"] = regroupement des résultats par état et type de licence

facet.mincount (numerique) – nombre minimum de résultats pour être comptabilisé comme facette (optionnel). Ne filtre pas les résultats de la recherche.

facet.limit (numerique) – nombre maximum de facettes pour un champ (optionnel). Ne filtre pas les résultats de la recherche.

Include_drafts (*booléen*) – Si la valeur est « Vrai », les jeux de données préliminaires seront inclus dans les résultats. Un utilisateur ne recevra que ses propres jeux de données et un sysadmin sera renvoyé à tous les jeux de données de brouillon. Facultatif, la valeur par défaut est « Faux ».

Include_private (*booléen*) – si « Vrai », les ensembles de données privés seront inclus dans les résultats. Seuls les ensembles de données privés provenant des organisations de l'utilisateur seront renvoyés et les administrateurs système recevront tous les ensembles de données privés. Facultatif, la valeur par défaut est « Faux ».

Exemples :

q=permis	jeux de données contenant le mot « permis »
fq=tags:économie	jeux de données avec l'étiquette « économie »
facet.field=["tags"] facet.limit=10 rows=0	liste des 10 étiquettes les plus utilisées
q=metadata_modified:["2017-06-01T00:00:00Z" TO *]	jeux de données modifiés depuis le 1er juin 2017

3.7 package_show

Renvoie un jeu de données (package) et ses jeux de données (resources).

Paramètres :

id (texte) – le nom système ou l'id d'un jeu de données

limit (entier) – le nombre maximal de jeux de données à renvoyer (facultatif)

Exemples :

```
http://[srv-data4citizen]/api/3/action/package_show?id=dataset-ventes
http://[srv-data4citizen]/api/3/action/package_show?id=46a09592-399a-4e2b-a3a9-7c22a501c7ef
```

Résultat :


```
{
  "help": "http://[srv-data4citizen]/api/3/action/help_show?name=package_show",
  "success": true,
  "result": {
    "license_title": "CC0 1.0",
    "maintainer": "",
    "relationships_as_object": [],
    "private": false,
    "maintainer_email": "",
    "num_tags": 3,
    "id": "46a09592-399a-4e2b-a3a9-7c22a501c7ef",
    "metadata_created": "2016-11-22T13:55:07.653647",
    "metadata_modified": "2017-06-13T14:00:48.824087",
    "author": "",
    "author_email": "",
    "state": "active",
    "version": "",
    "creator_user_id": "a8b1eff4-a80b-419c-afa3-3bdea0b15352",
    "type": "dataset",
    "resources": [
      {
        "cache_last_updated": null,
        "package_id": "46a09592-399a-4e2b-a3a9-7c22a501c7ef",
        "webstore_last_updated": null,
        "datastore_active": true,
        "id": "518aec96-def6-4183-b4d8-df93213ca0e6",
        "size": null,
        "state": "active",
        "hash": "",
        "description": "",
        "format": "CSV",
        "last_modified": "2017-06-02T08:38:08.840891",
        "url_type": "upload",
        "mimetype": null,
        "cache_url": null,
        "name": "Ventes par pays avec lat-long.csv",
        "created": "2016-11-22T14:55:08.539122",
        "url": "http://[srv-data4citizen]/dataset/46a09592-399a-4e2b-a3a9-
7c22a501c7ef/resource/518aec96-def6-4183-b4d8-df93213ca0e6/download/ventes-par-pays-avec-lat-long.csv",
        "webstore_url": null,
        "mimetype_inner": null,
        "position": 0,
```

```
        "revision_id": "068219be-1ed7-403e-8334-7c39c9c2dbfb",
        "resource_type": null
      }
    ],
    "num_resources": 1,
    "tags": [
      {
        "vocabulary_id": null,
        "state": "active",
        "display_name": "développement",
        "id": "8a22827b-ca1f-4da5-aa30-967981b785af",
        "name": "développement"
      },
      {
        "vocabulary_id": null,
        "state": "active",
        "display_name": "métropole",
        "id": "fb07dfe3-af7d-427f-8dd0-12a9e11e8c6f",
        "name": "métropole"
      },
      {
        "vocabulary_id": null,
        "state": "active",
        "display_name": "économie",
        "id": "e890ca09-1a73-4df0-86aa-87a18e49a930",
        "name": "économie"
      }
    ],
    "groups": [
      {
        "display_name": "Transports",
        "description": "Tout ce qui a trait à la question des transports",
        "image_display_url": "",
        "title": "Transports",
        "id": "7b0d1853-cf15-4a7d-94ce-1b9544859ada",
        "name": "transports"
      }
    ],
    "license_id": "CC0-1.0",
    "relationships_as_subject": [],
    "organization": {
      "description": "",
      "created": "2016-10-28T14:31:21.869272",
      "title": "bpm",
      "name": "bpm",
      "is_organization": true,
      "state": "active",
      "image_url": "",
      "revision_id": "34e0f074-714c-4e76-91a7-719dff0223d0",
      "type": "organization",
      "id": "135f9f15-69a6-41ee-a89f-e716c5a66a71",
      "approval_status": "approved"
    },
    "name": "dataset-ventes",
    "isopen": true,
    "url": "",
    "notes": "Extraction de Vanilla vers Data4citizen",
    "owner_org": "135f9f15-69a6-41ee-a89f-e716c5a66a71",
    "extras": [],
    "license_url": "https://creativecommons.org/publicdomain/zero/1.0/",
    "title": "Dataset Ventes",
    "revision_id": "51e6e597-b507-4775-bec1-049de0fcc07a"
  }
}
```

3.8 package_create

Créé un nouveau jeu de données (package).

La clé API d'un utilisateur possédant le droit de création de nouveau jeux de données doit être fournie. Renvoie le dataset complet au format JSON.

Paramètres :

name (texte) – le nom système du jeu de données. La valeur du paramètre doit être entre 2 et 100 caractères et n'utiliser que des valeurs alphanumériques en minuscules, des tirets haut (-) ou bas (_).

return_id_only (texte) – indique si seul l'identifiant du jeu de données créé doit être renvoyées. (optionnel) Valeur par défaut : 'False'

title (texte) – le titre du jeu de données (optionnel). Valeur par défaut : paramètre 'name'

author (texte) – le nom de l'auteur du jeu de données (optionnel).

author_email (texte) – l'adresse mail de l'auteur du jeu de données (optionnel).

maintainer (texte) – le nom du gestionnaire du jeu de données (optionnel).

maintainer_email (texte) – l'adresse mail du gestionnaire du jeu de données (optionnel).

license_id (texte) – l'identifiant de la licence du jeu de données (optionnel). Exemple : `fr-lo-2.0`

url (texte) – l'url de la source du jeu de données (optionnel).

version (texte) – la version du jeu de données (optionnel).

state (texte) – l'état du jeu de données (optionnel). Valeur par défaut : 'active'

type (texte) – le type du jeu de données (optionnel).

resources (texte) – les fichiers de données du jeu de données (optionnel). Voir la fonction API resource_create

tags (liste) – les « étiquettes » associées au jeu de données (optionnel). Liste de dictionnaires JSON

extras (liste) – les champs complémentaires du jeu de données (optionnel). Liste de dictionnaires JSON

groups (liste) – les groupes auxquels appartiennent le jeu de données identifiés par leur ID ou leur nom système (optionnel). Liste de dictionnaires JSON

owner_org (texte) – id de l'organisation propriétaire du jeu de données (optionnel).

Private (booléen) – Si « Vrai » crée un ensemble de données privé

Note (texte) – une description de l'ensemble de données (facultatif)

3.9 current_package_list_with_resources

Renvoie la liste des jeux de données avec les métadonnées des fichiers de données associés

La liste est triée par ordre chronologique inverse sur la date de modification.

Paramètres :

limit (numérique) – nombre de résultats renvoyés (optionnel)

offset (numérique) – point de départ de la pagination (/!\ : attention, les N premiers dataset sont numérotés de 0 à N-1, pour obtenir les suivants, offset doit être égal à N) (optionnel)

page (texte) – quand la limite est donnée, quelle page retourner, Obsolète: utilisation offset

3.10resource_search

Recherche sur les fichiers de données. Renvoie la liste des fichiers de données (resource) satisfaisant les critères de la requête.

Le résultat est organisé en deux « champs » : 'count' et 'results'. Le champ 'count' contient le nombre de ressources trouvées. Le champ 'results' est un dictionnaire JSON de métadonnées des fichiers de données.

Paramètres :

query (texte) – critères de recherche sous la forme « nom_de_champ:valeur » (**obligatoire**)

champ (dict des champs pour recherche des termes) – Obsolète

order_by (texte) – un champ sur sur le modèle de ressource qui ordonne les résultats.

offset (entier) – Applique un décalage à la requête.

Limit (entier) – applique une limite à la requête.

Exemples :

```
http://[srv-data4citizen]//api/3/action/resource_search?query=format:csv
```

```
http://[srv-data4citizen]//api/3/action/resource_search?query=name:vanilla
```

Résultat :


```
{
          "help": "http://[srv-data4citizen]/api/3/action/help_show?name=resource_search",
          "success": true,
          "result": {
                    "count": 1,
                    "results": [{
                              "key": "Vanilla KPI Miniatures Discounts.csv_20161117_092212",
                              "cache_last_updated": null,
                              "package_id": "0f8fa6f7-cc92-433f-9f47-ce5ec70fca40",
                              "webstore_last_updated": null,
                              "file": "FieldStorage('file', u'13883476.csv')",
                              "datastore_active": true,
                              "id": "fc9ed169-20b4-4499-980d-82aa7eee8ed1",
                              "size": null,
                              "state": "active",
                              "last_modified": "2016-11-17T08:18:07.716668",
                              "hash": "",
                              "description": "",
                              "format": "CSV",
                              "mimetype_inner": null,
                              "url_type": "upload",
                              "mimetype": null,
                              "cache_url": null,
                              "name": "Vanilla KPI Miniatures Discounts.csv",
                              "created": "2016-11-17T09:18:07.746970",
                              "url": "http://[srv-data4citizen]/dataset/0f8fa6f7-cc92-433f-9f47-
ce5ec70fca40/resource/fc9ed169-20b4-4499-980d-82aa7eee8ed1/download/13883476.csv",
                              "webstore_url": null,
                              "position": 0,
                              "revision_id": "f8d8e65e-1877-4c9f-bca3-28614d247030",
                              "resource_type": null
                    }
}
```

3.11resource_show

Renvoie les métadonnées d'un fichier de données (resource).

Paramètres :

id (texte) – l'identifiant d'un fichier de données (obligatoire)

include_tracking (booléen) – ajouter des informations de suivi à l'ensemble de données et aux ressources (par défaut: False)

Exemples :

```
http://[srv-data4citizen]/api/3/action/resource_show?id=_fc9ed169-20b4-4499-980d-82aa7eee8ed1
```

Résultat :


```
{
          "help": "http://[srv-data4citizen]/api/3/action/help_show?name=resource_show",
          "success": true,
          "result": {
                    "url_type": "upload",
                    "cache_last_updated": null,
                    "package_id": "0f8fa6f7-cc92-433f-9f47-ce5ec70fca40",
                    "webstore_last_updated": null,
                    "file": "FieldStorage('file', u'13883476.csv')",
                    "datastore_active": true,
                    "id": "fc9ed169-20b4-4499-980d-82aa7eee8ed1",
                    "size": null,
                    "state": "active",
                    "hash": "",
                    "description": "",
                    "format": "CSV",
                    "last_modified": "2016-11-17T08:18:07.716668",
                    "key": "Vanilla KPI Miniatures Discounts.csv_20161117_092212",
                    "mimetype": null,
                    "cache_url": null,
                    "name": "Vanilla KPI Miniatures Discounts.csv",
                    "created": "2016-11-17T09:18:07.746970",
                    "url": "http://[srv-data4citizen]/dataset/0f8fa6f7-cc92-433f-9f47-
ce5ec70fca40/resource/fc9ed169-20b4-4499-980d-82aa7eee8ed1/download/13883476.csv",
                    "webstore_url": null,
                    "mimetype_inner": null,
                    "position": 0,
                    "revision_id": "f8d8e65e-1877-4c9f-bca3-28614d247030",
                    "resource_type": null
          }
}
```

3.12resource_create

Ajoute un fichier de données au jeu de données spécifié.

Paramètres :

package_id (texte) – l'identifiant du jeu de données auquel ajouter le fichier (obligatoire)

url (texte) – l'url du fichier de données (obligatoire)

revision_id (texte) – la révision du fichier de données (optionnel)

description (texte) – la description du fichier de données (optionnel)

hash (texte) – le hash du fichier de données (optionnel)

name (texte) – le nom du fichier de données (optionnel)

resource_type (texte) – le type du fichier de données (optionnel)

mimetype (texte) – le type mime du fichier de données (optionnel)

mimetype_inner (texte) – le hash du fichier de données (optionnel)

size (numérique) – la taille du fichier de données (optionnel)

created (texte) – la date de création du fichier de données (optionnel) La chaîne de caractère doit être une date au format ISO

last_modified (texte) – la date de création du fichier de données (optionnel) La chaîne de caractère doit être une date au format ISO

upload (field storage) – envoi multipart/form-data du fichier de données (optionnel)

format (texte) – (Optionnel)

cahce_last_updated (iso date texte) – (Optionnel)

Exemple :

Utilisation de CURL pour la création d'un fichier de données et le chargement du fichier associé.

```
curl -H'Authorization: clé-api-data4citizen' 'http://[srv-
data4citizen]//api/action/resource_create' --form upload=@fichier_a_charger --form
package_id=id_du_jeu_de_donnee
```

3.13resource_update

Mise à jour du fichier de données spécifié.

Data4citizen considère qu'il s'agit d'une nouvelle version du fichier.

Paramètres :

id (texte) – l'id du fichier de données à mettre à jour (obligatoire)

Exemple :

Utilisation de CURL pour la mise à jour d'un fichier de données

```
curl -H'Authorization: clé-api-data4citizen' 'http://[srv-
data4citizen]//api/action/resource_update' --form upload=@fichier_a_charger --form
id=id_fichier_donnees
```

Requête HTTP POST pour la mise à jour d'un fichier de données

```
http --json POST http://[srv-data4citizen]/api/3/action/resource_update
id=<resource id> upload=@updated_file.csv Authorization:<api key>
```

4 Datastore

L'API du Datastore permet d'interagir directement avec la base de données dans laquelle sont stockés les jeux de données dans Data4citizen.

Il est possible notamment de télécharger directement un fichier de données (resource) depuis une URL formattée comme ceci :

```
http://[data4citizen-url][:port]/datastore/dump/[resource-id]
```

Le format de fichier par défaut est le **CSV** (Comma Separated Value).

Options de format de fichier :

Option	Description
?bom=true	CSV « Excel-Compatible »

Exemple :

```
http://[srv-data4citizen]/datastore/dump/518aec96-def6-4183-
b4d8-df93213ca0e6?bom=true
```

L'API du Datastore permet aussi des opérations directes en base sur les fichiers de données :

- Création
- Suppression
- Mise à jour
- Information sur la structure du fichier de données.

Les fonctions de l'API du Datastore s'exécutent en POST avec les options de commandes passées sous la forme d'un fichier JSON associé à l'appel.

4.1 datastore_info

Renvoie des informations (noms de colonnes et types, nombre d'enregistrements) sur les données chargées pour un fichier de données.

Paramètre :

id (UUID) – l'identifiant du fichier de données (resource id)

```
curl -H "Content-Type: application/json" -X POST  --data '{ "id": "518aec96-def6-
4183-b4d8-df93213ca0e6"}' http://[srv-data4citizen]/api/3/action/datastore_info
```

Résultat :


```
{
        "help": "http://[srv-data4citizen]/api/3/action/help_show?name=datastore_info",
        "success": true,
        "result": {
                "meta": {
                        "count": 22723
                },
                "schema": {
                        "orderfact_totalprice": "number",
                        "country_longitude": "number",
                        "country_name_country": "text",
                        "country_latitude": "number",
                        "orderfact_status": "text"
                }
        }
}
```

4.2 datastore_create

Ajoute une nouvelle table au datastore.

L'action datastore_create permet d'envoyer des données au format JSON afin qu'elles y soient stockées pour un fichier de données (resource)

Cette action peut être appelée autant de fois que nécessaire pour insérer des données, ajouter des champs, changer les alias ou les indexes, de même que les clés primaires.

Pour créer une resource datastore vide et un fichier de données (resource) en une seule opération, omettre le paramètre 'resource_id'

Pour créer une resource depuis le contenu d'un fichier, il faut fournir une URL valide dans le dictionnaire JSON du paramètre 'resource'.

Paramètres :

resource_id (texte, UUID) – l'id du fichier de données (resource) pour lequel créer la table du datastore.

force (booléen) – mettre à 'True') pour modifier une resource en lecture seule

resource (dictionnaire) – dictionnaire JSON qui sera passé à l'action API resource_create pour créer les métadonnées du fichier de données associé à la table du datastore (optionnel)

aliases (liste de dictionnaires) – nom(s) des aliases en lecture seule de la resource

fields (liste de dictionnaires) – champs (colonnes) et leurs métadonnées (optionnel)

Un champ est défini par son nom et le type de données de la manière suivante :

```
{
    "id":    # une chaîne spécifiant le nom du champ
    "type":  # le type de données du champ
}
```

Exemple :

```
[
    {
        "id": "champ1",
        "type": "int4"
    },
    {
        "id": "champ2"
        # le type est optionnel
    }
]
```

records (liste de dictionnaires) – les données à insérer dans la table du datastore

Un enregistrement est défini de la manière suivante :

```
{
    "champ1":  # données
    "<nom_champ>":  # données
    # etc
}
```

Exemple (2 enregistrements)

```
[
    {
        "champ1": 100,
        "texte1": "Un champ contenant du texte"
    },
    {
        "champ1": 42
    }
]
```

primary_key (liste ou chaines de caractères séparées par des virgules) – les champs constituant la clé unique de la table (optionnel)

4.3 datastore_upsert

Insertion ou mise à jour des données dans une table du datastore.

Pour que les méthodes « upsert » ou « update » fonctionnent, une clé unique doit avoir été définie grâce à l'action datastore_create.

Description des méthodes :

UPSERT : mise à jour si un enregistrement avec la même clé existe. Dans le cas contraire les données sont insérées. CLE PRIMAIRE OBLIGATOIRE

INSERT : Insertion uniquement. Cette méthode est plus rapide que l'upsert mais elle échouera si une clé unique est en double.

UPDATE : mise à jour uniquement. Une exception se produira si une clé devant être mise à jour est inexistante. CLE PRIMAIRE OBLIGATOIRE

Paramètres :

resource_id (texte, UUID) – l'id du fichier de données (resource) qui sera mis à jour.

force (booléen) – mettre à 'True') pour modifier une resource en lecture seule

records (liste de dictionnaires) – les données à insérer dans la table du datastore

method (texte) – méthode à utiliser pour l'insertion de données dans la table du datastore. Les options possibles sont : upsert, insert, update. Valeur par défaut : upsert