Projet Jeux Olympiques - Journal de bord

T. Burnel, N. Grim, M. Griveau, M. Mechentel

Janvier 2024

Introduction

Nous sommes une équipe de datajournalistes chargée, à l'approche des Jeux Olympiques de Paris, d'étudier l'existence ou non de liens entre le succès d'un pays aux Jeux et sa richesse. Notre hypothèse de départ est la suivante : les pays développés gagnent significativement plus de médailles en raison de leur niveau d'investissement dans le domaine sportif.

À l'aune de l'examen et du traitement des données que nous avons à notre disposition, nous avons pour ambition de déterminer si la politique d'investissement dans les infrastructures sportives des pays a une quelconque influence sur leurs résultats aux épreuves des Jeux.

Notre travail sera divisé en deux temps. Nous mettrons en relief le nombre de médailles remportées par les pays lors de six éditions des Jeux Olympiques (1996, 2000, 2004, 2008, 2012, 2016) par rapport à leur population, à leur Produit Intérieur Brut et au pourcentage de ce dernier investi dans le domaine sportif. Un second temps sera consacré à l'examen de la réussite de la France et du Royaume-Uni aux épreuves de natation sur la base du nombre d'infrastructures relatives pour 100.000 habitants sur leur territoire.

Le choix de ces pays repose sur divers points de convergence : leur nombre d'habitants, leur économie et leur qualité de pays organisateur des Jeux.

Travailler sur une période de vingt ans nous semble suffisant pour cerner ou non le début d'une tendance économique vis-à-vis de la réussite des pays aux Jeux Olympiques. Le temps et les ressources dont nous disposons tendent à confirmer le choix de cette période – suffisamment éloquente en données pour nous permettre de travailler avec un matériau d'une qualité et d'une quantité honorables sans qu'il ne faille faire entorse à nos capacités financières – strictement égales à zéro – et humaines – strictement égales à quatre.

Nous sommes *ipso facto* conscients de la prudence dont nous devrons faire preuve, le moment des conclusions venu. Il s'agit bien davantage d'un travail préliminaire que la rigueur journalistique force à éprouver et à approfondir que d'une parole d'évangile dispensable de toute remise en question.

Le résultat du traitement des données, réalisé sur le logiciel Dataiku, constituera une base à la conception de datavisualisations sur le logiciel Tableau Public et d'une application web exploitables pour l'écriture d'articles journalistiques sur le sujet des Jeux Olympiques. Les données mises à disposition, exception faite du nom des régions de France, sont en langue anglaise dans l'idée d'une exploitation par la presse internationale.

Jeux de données

2.1 Jeux du premier flux

Notre premier flux concerne les médailles remportées par les pays lors des six éditions des Jeux Olympiques sur lesquelles nous avons décidé de travailler. En amont du traitement, nos jeux étaient au nombre de deux. Le premier présente le nombre de médailles remportées par sportif aux Jeux sur une période de 120 ans. [1] Le deuxième, le Produit Intérieur Brut des pays et son pourcentage investi dans le domaine sportif. [2]

Nous avons enrichi ces jeux grâce au résultat d'une requête SPARQL, saisie sur le service de Wikidata. Le jeu ainsi produit présente le nombre d'habitants de l'ensemble des pays répertoriés dans la base de données de Wikidata, sur une période de trente ans (1993 - 2023). [3]

Trois autres jeux ont dû faire leur entrée au cours du traitement pour pallier de nombreux manquements. Nous avons constaté qu'en dépit d'indiquer

1. *120 years of Olympic history : athletes and results*. URL : `https://www.kaggle.com/datasets/heesoo37/120-years-of-olympic-history-athletes-and-results` (visité le 17/01/2024).
2. *Download entire World Economic Outlook database, April 2019*. IMF. URL : `https://www.imf.org/en/Publications/WEO/weo-database/2023/October/download-entire-database` (visité le 17/01/2024).
3. *Wikidata Query Service*. URL : `https://w.wiki/8uTN` (visité le 27/01/2024).

le pourcentage du Produit Intérieur Brut investi dans le domaine sportif, le jeu du Fonds Monétaire International ne contenait pas le PIB des pays. Nous avons introduit un jeu de la Banque mondiale [4] dans le flux – lequel contient le montant du PIB avec une profondeur chronologique suffisante.

Les deux autres jeux concernent un problème de coordonnées. Lorsque Dataiku identifie des données de type *country*, nous pensions pouvoir en récupérer les coordonnées – notamment afin de les exploiter pour les datavisualisations. L'extension *reverse geocoding plugin* aurait pu nous le permettre, sans succès. Nos suppositions portent sur notre version gratuite de Dataiku et de son nombre réduit de fonctionnalités.

Ne disposant d'aucun budget pour bénéficier d'une version payante, nous avons recherché un jeu contenant les coordonnées des pays et l'avons trouvé sur la plateforme GitHub. [5] Nonobstant, les longitudes et latitudes des pays ainsi récupérées se sont avérées être fautives le moment des datavisualisations venu. C'est pourquoi nous avons mis au point une deuxième SPARQL, capable de renvoyer les bonnes coordonnées de tous les pays présents dans la base de Wikidata. [6] Le jeu ainsi récupéré correspond au dernier renfort dont nous avons eu besoin pour être en mesure de réaliser des datavisualisations.

2.1.1 Jeu de Kaggle

Le jeu est composé de 15 colonnes et 271.117 lignes. 7 colonnes fournissent des informations sur les athlètes (*ID*, *Name*, *Sex*, *Age*, *Height*, *Weight*, *Me-*

4. *World Bank Open Data*. World Bank Open Data. URL : `https://data.worldbank.org` (visité le 20/01/2024).

5. Alexandru-Paul Copil. *Countries codes and coordinates*. URL : `https://gist.github.com/cpl/3dc2d19137588d9ae202d67233715478` (visité le 25/01/2024).

6. *Wikidata Query Service*. URL : `https://w.wiki/8yne` (visité le 28/01/2024).

dal) ; 2 colonnes sur leurs pays (*Team*, *NOC*) et 6 sur les éditions des Jeux Olympiques (*Games* ; *Year* ; *Season* ; *City* ; *Sport* ; *Event*).

Les données sont exprimées en langue anglaise. La casse des chaînes de caractères suit le schéma d'une majuscule à chaque nouveau mot – exception faite des unités de mesure dans la colonne *Event* (comprendre *Épreuve*) : le terme « metres » est toujours noté en bas-de-casse. 2 colonnes, *Games* et *Year* contiennent des données de date, exprimées en année. La colonne *Games* a ceci de particulier qu'elle agrège une donnée de date et une donnée textuelle. Les Jeux Olympiques d'été de 1992 sont ainsi notés « 1992 Summer ».

Le jeu ne présente pas de défauts majeurs sur les colonnes que nous comptons conserver lors du traitement. Un problème plus préoccupant concerne la dénomination des pays, parfois obsolète (« Soviet Union », « West Germany ») ou impropres (« Japan-1 », « Switzerland-2 »). Nous effectuerons les jointures sur le code NOC des pays (un code unique attribué par le Comité National Olympique) pour plus de fiabilité.

2.1.2 Jeu du FMI

Le jeu est composé de 42 colonnes pour 7.715 lignes. La colonne *Country Name* permet d'identifier les pays. Celle nommée *COFOG Function Name* correspond au secteur des investissements et a pour unique valeur *Expenditure on recreational & sporting services*. La colonne *Unit Name* précise l'échelle de calcul des investissements (soit en pourcentage du PIB, *Percent of GDP*, soit en monnaie courante *Domestic currency*). Le montant des investissements est renseigné sur 30 colonnes, une par année de 1993 à 2022.

Les 9 autres colonnes correspondent à des codes et des dénominations

propres au FMI (*Country Code*, *COFOG Function Code*, *Sector Name*, *Sector Code*, *Unit Code*, *Attribute*, *Indicator Code*, *Global DSD Time Series Code*, *col_41*). Le potentiel de croisement avec d'autres jeux étant par la force des choses très limité, nous n'utiliserons pas ces données.

L'ensemble du jeu est rédigé en langue anglaise. Noms de pays, sigles, codes, acronymes et unités de mesure (notées « Cash », « Non Cash », « Gross », « Net ») à part, la casse des données textuelles implique la présence d'une seule majuscule en début de chaîne. Nous notons l'utilisation de caractères spéciaux, à l'instar de l'esperluette ou de la barre oblique.

Exception faite des *Country Code* et du code correspondant à la valeur « Domestic currency », l'intégralité des codes sont un mélange de caractères numériques et textuels en haut-de-casse (« S13112 » pour la colonne *Sector Code* ou « XDC_R_B1GQ » dans la colonne *Unit Code*).

Les noms de pays souffrent d'un problème d'harmonisation et d'actualité (« Russian Federation », « China, P.R. : Mainland », etc.).

Ce jeu est de loin le moins bien structuré et le plus fautif de notre *set* basal. Bien que nous ne comptions pas les conserver au cours du traitement, les trois dernières colonnes (*Indicator Code*, *Global DSD Time Series Code*, *col_41*) sont d'une opacité confondante et témoignent du problème majeur du jeu : énormément de données sont vides, nulles ou incompréhensibles (« GERS_G14_GDP_PT » pour la colonne *Indicator Code*, « A|GB|S1311|W0|S1|G2M|_Z|_Z|GF0801| XDC|_T|_X » pour la colonne *Global DSD Time Series Code*).

Également, au sein des colonnes correspondant au montant des investissements par année se côtoient des lignes vides et des sigles (« NA », « NP »,

« AC »), sans justification ni harmonisation aucune. En somme, ce jeu représente un véritable enjeu de nettoyage et de croisement des données.

2.1.3 Jeu de Wikidata (population)

En préambule de l'analyse du jeu de Wikidata, revenons un instant sur l'élaboration de la requête nous ayant permis de l'obtenir. Celle-ci renvoie le nombre d'habitants de l'ensemble des pays présents dans la base de données de Wikidata sur une période de trente ans (1993 - 2023) :

```
SELECT ?paysLabel ?population ?date ?cio
WHERE
{
 ?pays wdt:P31 wd:Q6256.
 ?pays wdt:P984 ?cio.
 ?pays p:P1082 ?populationStatement.
 ?populationStatement ps:P1082 ?population.
 ?populationStatement pq:P585 ?date.
 FILTER(YEAR(?date) >= (YEAR(NOW()) - 30)).
 SERVICE wikibase:label { bd:serviceParam wikibase:language
     "[AUTO_LANGUAGE],fr". }
}
ORDER BY ?paysLabel ?date
```

Être en mesure de requêter l'intégralité des pays a été la première étape de la construction de notre requête. Le premier triplet permet de spécifier la nature – notée `wdt:P31` – de notre variable inconnue `?pays` en la faisant correspondre à l'objet `country` – noté `wd:Q6256` :

```
?pays wdt:P31 wd:Q6256.
```

Le deuxième triplet constitue un ajout tardif, destiné à faciliter les jointures avec les autres jeux de données. Comme nous travaillons sur les Jeux Olympiques, les pays peuvent être non seulement identifiés par les codes basés sur la norme ISO 3166 mais aussi par les codes du CIO (Comité International Olympique). Nous avons donc récupéré cette donnée grâce à la propriété Wikidata correspondante – notée `wdt:P984` :

```
?pays wdt:P984 ?cio.
```

La deuxième partie de notre requête renvoie le nombre d'habitants par pays en prenant en compte une dimension chronologique. La complexité de cette demande requiert un parcours de graphique en quatre temps.

Pour ce faire, nous avons consulté la liste de préfixes [7] de Wikidata afin de créer une nouvelle variable : `?populationStatement`. Le troisième triplet a recours à la classe `population` – notée `p:P1082` – et signifie que la variable `?populationStatement` a pour valeur la population des pays :

```
?pays p:P1082 ?populationStatement.
```

L'obtention du nombre d'habitants a nécessité l'utilisation du préfixe `ps`, lequel permet de récupérer la valeur de la propriété relative à la population :

```
?populationStatement ps:P1082 ?population.
```

Enfin, nous avons rédigé le dernier triplet sur la base du préfixe `pq`

7. *Wikidata prefixes (E49) - Wikidata.* URL : `https://www.wikidata.org/wiki/EntitySchema:E49` (visité le 20/01/2024).

pour attribuer la valeur chronologique à la variable `?date`. Un filtre y a été appliqué pour exprimer les limites extrêmes de notre période, soit `NOW` pour 2023 et `-30` pour 1993 :

```
?populationStatement pq:P585 ?date.
FILTER(YEAR(?date) >= (YEAR(NOW()) - 30)).
```

La première et la dernière ligne permettent de cadrer l'affichage des résultats. Lorsque la requête s'exécute, Wikidata renvoie le nom de chaque pays (`?paysLabel`) par ordre alphabétique, le nombre d'habitants (`?population`) et l'année correspondante (`?date`) par ordre croissant :

```
SELECT ?paysLabel ?population ?date
ORDER BY ?paysLabel ?date
```

Nous nous sommes heurtés à plusieurs obstacles avant de rendre la requête fonctionnelle. Il nous a fallu un certain temps avant de comprendre la nécessité d'un parcours de graphique en quatre temps et du recours à une variable telle que `?paysStatement`. À cet égard, la documentation sur l'utilisation des propriétés `population` [8] et `date` [9] a été d'un grand secours.

Nous avons également pris en exemple la requête *Population in Europe after 1960*. [10] La consulter a été l'occasion d'une meilleure compréhension des propriétés Wikidata ainsi qu'une porte ouverte à la lecture de la documentation et à l'appréhension des requêtes en deux temps (collecte des propriétés

8. *Population*. URL : `https://www.wikidata.org/wiki/Property:P1082` (visité le 20/01/2024).
9. *Point in time*. URL : `https://www.wikidata.org/wiki/Property:P585` (visité le 20/01/2024).
10. *Wikidata Query Service*. URL : `https://w.wiki/8uHH` (visité le 20/01/2024).

d'une classe puis requêtage en fonction de notre besoin).

La rédaction de cette première requête a été d'une grande aide pour les autres recours à Wikidata qui ont émaillé notre chaîne de traitement.

Le jeu ainsi généré est composé de 13 colonnes et 3.737 lignes. Les lignes de la colonne, *paysLabel.xml :lang* sont composées d'une même et unique donnée : « fr ». Il s'agit de langue dans laquelle les noms des pays sont renvoyés.

Sur les 12 colonnes restantes, 6 fonctionnent selon une logique de duo et 6 autres selon une logique de trio. Pour chaque donnée, le jeu indique sont *type*, sa *value* et, pour les trio, son *datatype*. Ainsi, la donnée « cio » fait l'objet de 2 colonnes : *cio.type*, *cio.value* (*idem* pour les données « isoCode » et « paysLabel ») et la donnée « date » fait l'objet de 3 colonnes : *date.datatype*, *date.type*, *date.value* (*idem* pour la donnée « population »).

Bien que notre requête ne précise vouloir recueillir ni le *type* ni le *datatype* des données, ces colonnes sont malgré tout présentes dans le jeu de sortie. Les données textuelles sont exprimées en anglais (mis à part les noms de pays) et sont en bas-de-casse – exception faite des sigles et des noms de pays. Les dates sont exprimées selon le schéma YYYY-MM-DDThh:mm:ssZ. L'intégralité des *types* correspondent à « literal ». Les colonnes *datatype* ont pour valeur des liens renvoyant à des schémas XML du W3C.

Le jeu est d'une propreté exemplaire, les données sont harmonisées et correctement typées, ce qui nous sera d'une grande aide lors des jointures.

2.1.4 Jeu de la Banque mondiale

Le jeu se compose de 68 colonnes pour 533 lignes. Chaque ligne pleine est en réalité séparée de la suivante par une ligne vide. De surcroît, la dernière colonne est intégralement vide et ne porte que le sommaire nom de *col_67*. Sur l'ensemble du jeu, 63 colonnes correspondent aux années pour lesquelles le PIB des pays est renseigné (1960 - 2022). 2 autres colonnes donnent des informations sur les pays : leur nom, leur code (*Country Name*, *Country Code*) et les 2 restantes, sur la donnée économique – en l'occurrence le PIB (*Indicator Name*, *Indicator Code*), en dollars courant.

Les données textuelles sont en langue anglaise. Les noms des pays portent une majuscule initiale. Les codes sont exclusivement en haut-de-casse. Les données numériques sont exprimées en décimaux.

Notons que certaines entrées de la colonne *Country Name* témoignent d'une certaine originalité dans la mesure où ils ne sont pas références à des pays nommément explicites (« Heavily indebted poor countries (HIPC) », « IBRD only », « Low & middle income », etc.). Ces excentricités soulignent l'importance des codes des pays pour effectuer les jointures et faire l'économie des entrées impropres. Enfin, les premières décennies souffrent d'un cruel manque de données mais plus nous avançons dans le temps, plus le jeu est complet sur l'ensemble des pays.

2.1.5 Jeu de GitHub

Le jeu se compose de 6 colonnes et 263 lignes. 2 de ces colonnes correspondent à la latitude et longitude (*Latitude (average)*, *Longitude (average)*)

des pays dont le nom nous est donné dans la colonne *Country*. Les trois autres colonnes (*Alpha-2 code*, *Alpha-3 code*, *Numeric code*) correspondent à des codes de pays prévus par la norme ISO 3166-1.

Les données textuelles sont exprimées en langue anglaise. Nonobstant, nous comptons nous appuyer sur les codes des pays pour réaliser les jonctions. Le jeu ne présente aucune donnée manquante.

2.1.6 Jeu de Wikidata (coordonnées)

Revenons derechef sur le détail de notre requête SPARQL, rédigée pour nous permettre de récupérer les coordonnée des pays recensés dans la base de données de Wikidata :

```
SELECT DISTINCT ?paysLabel ?geopoint ?noc
WHERE
{
 ?pays wdt:P31 wd:Q6256.
 ?pays wdt:P984 ?noc.
 ?pays wdt:P625 ?geopoint
 SERVICE wikibase:label { bd:serviceParam wikibase:language
     "[AUTO_LANGUAGE],en". }
}
```

Nous indiquons vouloir récupérer les données selon les colonnes suivantes : le nom du pays (noté `?paysLabel`), leurs coordonnées (notées `?geopoint`) et leur code CIO (noté `?noc`) :

```
 SELECT DISTINCT ?paysLabel ?geopoint ?noc
```

Le premier triplet est *stricto sensu* le même que celui de notre première requête Wikidata sur les populations (*cf. supra* p. 7).

Il permet de signifier que notre variable inconnue `?pays` est de nature (notée `wdt:P31`) « pays » (noté `wd:Q6256`) :

```
?pays wdt:P31 wd:Q6256.
```

Le deuxième triplet est également tout à fait similaire à son homonyme de la requête sur les populations (*cf. supra* p. 7). Il assigne à la variable `?noc` les codes des pays prévus par le Comité Olympique International grâce à la propriété relative notée `wdt:P984` :

```
?pays wdt:P984 ?noc.
```

Enfin, le dernier triplet récupère les coordonnées géographiques des pays grâce à la propriété correspondante, notée `wdt:P625` :

```
?pays wdt:P625 ?geopoint
```

Il en ressort un jeu composé de 8 colonnes et 185 lignes. Il présente la même logique de duo et de trio que notre autre jeu Wikidata (*cf. supra* p. 10) : les données « noc » et « paysLabel » font l'objet de 2 colonnes et les coordonnées, « geopoint », font l'objet de 3 colonnes. La longitude et la latitude sont regroupées au sein de la même colonne, *geopoint.value* et sont présentées comme suit : « Point(<longitude> <latitude>) ». L'Islande a ainsi pour coordonnées « Point(-19.0 65.0) ».

Cette requête ayant été saisie dans l'urgence de devoir déboguer la réalisation des datavisualisations et la rédaction du présent journal de bord à

moins d'une semaine de la date butoir, les noms de pays sont exprimés en langue anglaise et les résultats n'ont pas été ordonnés par ordre alphabétique à l'instar de la requête sur la population. Nonobstant, la forme des données qui nous préoccupent – en l'occurrence, les codes NOC et les coordonnées – ne dépend pas de la langue pourvu que l'alphabet soit le même.

2.2 Jeux du second flux

Notre deuxième flux repose sur une comparaison entre la France et le Royaume-Uni à l'endroit de la natation. En amont du traitement, nos jeux étaient au nombre de cinq : trois pour le Royaume-Uni, deux pour la France.

Sur le Royaume-Uni, un premier jeu résulte d'une requête SPARQL, laquelle génère un jeu listant le nom des régions du pays, leur nombre d'habitants et leurs coordonnées. [11] Les deux autres jeux proviennent du même site, [12] l'un situe les infrastructures sportives par rapport à leur région, l'autre les décrit d'un point de vue davantage technique.

Sur la France, l'un des deux jeux est également une requête SPARQL, renvoyant là aussi le nom des régions du pays, leur nombre d'habitants et leurs coordonnées. [13] Le second jeu, fourni par le ministère de l'Éducation nationale, de la Jeunesse, des Sports et des Jeux Olympiques et Paralympiques [14] présente les infrastructures sportives et les relie aux régions françaises.

Au cours du traitement, nous avons réalisé qu'il nous fallait relier tôt

11. *Wikidata Query Service*. URL : `https://w.wiki/8yqy` (visité le 28/01/2024).
12. *Active Places Power*. URL : `https://www.activeplacespower.com/` (visité le 20/01/2024).
13. *Wikidata Query Service*. URL : `https://w.wiki/8ynm` (visité le 28/01/2024).
14. *Data ES - Base de données*. URL : `https://equipements.sports.gouv.fr/explore/dataset/data-es/information/` (visité le 17/01/2024).

ou tard ces divers jeux aux médailles remportées par les deux pays. C'est pourquoi nous avons fait usage d'une version du jeu de Kaggle issu de notre premier flux, où se trouvent les données nécessaires.

2.2.1 Jeu de Wikidata (régions de France)

Cette requête interroge la base de données de Wikidata selon le prisme des régions. Nous voulons ici afficher leur nom (`?RegionLabel`), leur nombre d'habitants (`?population`) et leurs coordonnées (`?geopoint`) :

```
SELECT ?RegionLabel ?population ?geopoint
WHERE
{
 ?Region wdt:P31 wd:Q36784.
 ?Region wdt:P1082 ?population.
 ?Region wdt:P625 ?geopoint.
 SERVICE wikibase:label { bd:serviceParam wikibase:language
     "[AUTO_LANGUAGE],fr". }
}
ORDER BY ?Region
```

Le premier triplet nous permet de faire correspondre à la variable `?Region` l'objet « région de France » noté `wd:Q48091` , derechef grâce à la propriété `wdt:P31` nous permettant de préciser la nature de notre sujet :

```
 ?Region wdt:P31 wd:Q36784.
```

Le deuxième triplet assigne à la variable `?population` la valeur de la

propriété `wdt:P1082`, c'est-à-dire la population – ici des régions :

```
?Region wdt:P1082 ?population.
```

Selon la même logique, le dernier triplet récupère les coordonnées des régions grâce à la variable `?geopoint` et à la propriété « coordonnées géographiques » notée `wdt:P625` :

```
?Region wdt:P625 ?geopoint.
```

Et nous trions les résultats selon l'ordre alphabétique du nom des régions :

```
ORDER BY ?Region
```

Le jeu ainsi généré est composé de 9 colonnes et 18 lignes. Nous n'expliciterons pas de nouveau les logiques d'atomisation des données en deux ou trois colonnes (*cf. supra* p. 10). Le jeu présente le nom des régions (*RegionLabel.type*), leur nombre d'habitants (*population.value*), leurs coordonnées (*geopoint.value*) et ce en français.

2.2.2 Jeu du ministère

Le jeu est composé de 104 colonnes et 143.192 lignes. Par souci de concision, nous ne nous attarderons que sur les colonnes que nous comptons conserver au cours du traitement. Ces dernières sont au nombre de quatre : *Type d'équipement sportif*, *Longitude (WGS84)*, *Latitude (WGS84)* et *Région Nom.*

La colonne *Type d'équipement sportif* représente un véritable enjeu de nettoyage. La nomenclature est très libre, les dénominations sont d'apparence toutes en bas-de-casse avec une majuscule initiale mais plusieurs excep-

tions perturbent ce schéma (« Parcours Acrobatique en Hauteur/Site d'accrobranche », « Structure Artificielle d'Escalade »).

De surcroît, nombre d'entrées font l'objet de multi-nommage par l'intermédiaire de la barre oblique (« Multisports/City-stades », « Salles polyvalentes / des fêtes / non spécialisées », « Piste d'aérodrome / d'aéroport », etc.) et d'autres sont accompagnées de précisions entre parenthèses : « Salle multisports (gymnase) », « Aire mixte (décollage et atterissage) », etc.

Les noms de régions sont saisies en bas-de-casse avec conservation des majuscules initiales. Enfin, la longitude et la latitude se basent sur le système *WGS 84 – World Geodetic System 1984*, un système géodésique mondial comprenant, entre autres, un modèle pour l'expression de coordonnées.

2.2.3 Jeu de Wikidata (régions d'Angleterre)

Cette requête est l'identique stricte de celle sur les régions de France (*cf. supra* p. 15). La description des triplets sera d'autant plus brève. Nous voulons afficher le nom des régions du Royaume-Uni (`?RegionLabel`), leur nombre d'habitants (`?population`) et leurs coordonnées (`?geopoint`) :

```
SELECT DISTINCT ?RegionLabel ?population ?geopoint
WHERE
{
 ?Region wdt:P31 wd:Q48091.
 ?Region wdt:P1082 ?population.
 ?Region wdt:P625 ?geopoint.
 SERVICE wikibase:label { bd:serviceParam wikibase:language
     "[AUTO_LANGUAGE],en". }
```

```
}
```

Le premier triplet fait correspondre la variable `?Region` à l'objet « région d'Angleterre » (`wd:Q48091`), grâce à la propriété « région » (`wdt:P31`) :

```
?Region wdt:P31 wd:Q48091.
```

Le deuxième triplet assigne à la variable `?population` la population (`wdt:P1082`) de notre sujet, c'est-à-dire les régions (`?Region`). Cependant, la seule propriété existante à cet égard concerne les régions d'Angleterre, non pas du Royaume-Uni. Cela aura des conséquences sur nos datavisualisations car nos autres données concernent bel et bien le Royaume-Uni.

```
?Region wdt:P1082 ?population.
```

Le dernier triplet récupère les coordonnées (`wdt:P625`) des régions (`?Region`) et les assigne à la variable `?geopoint` :

```
?Region wdt:P625 ?geopoint.
```

Il en ressort un jeu composé de 9 colonnes et 36 lignes. Le principe d'atomisation des données en deux ou trois colonnes est de nouveau à l'œuvre (*cf. supra* p. 10). Le jeu présente strictement les mêmes colonnes que le jeu résultant de la requête sur les régions de France : le nom des régions (*RegionLabel.type*), leur nombre d'habitants (*population.value*) et leurs coordonnées (*geopoint.value*) et ce en langue anglaise.

2.2.4 Jeux de *Sport England*

Le premier jeu provenant de *Sport England*, nommé *Geographics* est composé de 25 colonnes et 118.655 lignes. Comme son nom le suggère, les données présentées sont relatives aux infrastructures sportives du Royaume-Uni. Chacune possède un identifiant unique, répertorié dans la colonne *FACILITYID* et est rattachée à une région, grâce à la colonne *Region Name*. Leur nom est en anglais et en bas-de-casse, avec conservation des majuscules initiales. Les 2 autres colonnes qui survivront au traitement sont *Latitude* et *Longitude*.

Les 21 colonnes restantes fournissent une plus grande quantité de données sur la situation géographique et administrative des infrastructures – selon un schéma soit d'identifiants (*SiteID*), soit de code (*Output Area Code*), soit de colonnes en duo avec d'une part les noms (*Ward Name*, *Local Authority Name*, etc.), d'autre part les codes (*Ward Code*, *Local Authority Code*, etc.).

Le système de nommage et de dénomination ne laisse pas place à l'opacité. Certaines colonnes manquent cruellement de données (*Metro Name* ; *Core City Name* ; *LDP Code* ; *LDP Name*) mais nous ne comptons pas les conserver le moment du traitement venu.

Le deuxième jeu, nommé *SwimmingPool*, est composé de 24 colonnes et 6.474 lignes. Exception faite de la première colonne, *FacilityID*, l'intégralité des données est à propos de détails techniques et matériels sur les infrastructures (*Length* ; *Maximum Depth* ; *Movable Floor* ; *Seating*, etc.).

Toutes les données sont de type numérique – soit des entiers, soit des décimaux. Certaines colonnes (*Designated Accessible Provision*, *Designated*

Accessible Provision - Covered, *Designated Accessible Provision - Uncovered*, *Seating - Covered*, *Seating - Uncovered*, *Standing*, *Standing - Covered*, *Standing - Uncovered*) ont la singularité de faire se côtoyer des entrées vides et des entrées égales à zéro. Cependant, seule la colonne *FacilityID* nous importe afin de réaliser une jointure avec le jeu *Geographics*.

2.2.5 Jeu de Kaggle

Cette version du jeu *Kaggle* issu de notre premier flux est composée de 26 colonnes et 780 lignes. 2 colonnes, *NOC* et *Sport* correspondent respectivement aux codes CIO des pays et à l'intitulé du sport (en langue anglaise et bas-de-casse, avec conservation des majuscules initiales) ayant fait l'objet d'au moins épreuve aux Jeux Olympiques.

Les 24 autres colonnes correspondent aux six éditions des Jeux selon le schéma suivant : chaque année fait l'objet de 4 colonnes. 3 pour les types de médailles (bronze, argent, or) et 1 pour la somme de l'ensemble des médailles. Ainsi, les colonnes de l'année 2008 sont nommées comme suit :

2008 - Bronze, *2008 - Silver*, *2008 - Gold*, *2008 - Médailles*.

Traitement des données

3.1 Objectif du traitement

À l'aune de notre approche, notre premier flux a pour ambition de faire ressortir les données suivantes : les codes des pays (attribués par le *National Olympic Committee* – que nous nous permettrons d'abréger « NOC » ou « CIO ») permettant de les identifier indépendamment des barrières de langue, le nombre de médailles remportées aux six éditions des Jeux Olympiques ainsi que le pourcentage d'investissement des pays dans le domaine sportif par rapport à leur PIB et à leur population.

Le second flux se concentrera sur les médailles remportées par la France et le Royaume-Uni pour les épreuves de natation. Nous mettrons ces données en parallèle du nombre d'infrastructures sportives relatives recensées dans ces deux pays et évaluer, en somme, le nombre d'infrastructures pour 100.000 habitants dans chaque région.

3.2 Chaîne de traitement du premier flux

3.2.1 Préparation du jeu de la Banque mondiale

Le jeu de la Banque mondiale, nommé *GDP* dans Dataiku, correspond à cette partie du flux :

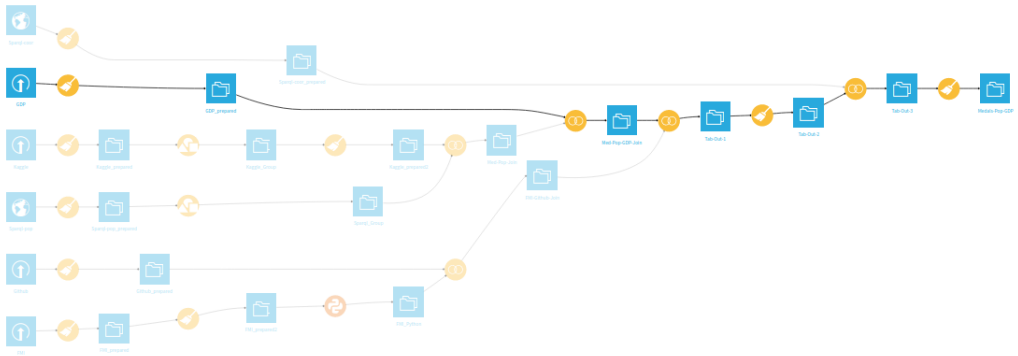


FIGURE 3.1 – Chaîne de traitement du jeu *GDP*

La première recette appliquée est une préparation. Elle consiste en la suppression de toutes les lignes vides – soit une ligne sur deux – et la conservation de seulement 7 colonnes sur 68 : *Country Code* pour les codes NOC (fondamentaux pour réaliser de futures jointures) et les 6 colonnes correspondant aux éditions des Jeux Olympiques couvertes par notre travail.

Les colonnes ainsi sélectionnées ont été renommées : *Country Code* devient *NOC* et les colonnes des années suivent le même motif : *1996* devient *1996 - GDP* (pour *Gross Domestic Product*, c'est-à-dire le Produit Intérieur Brut). Enfin, toutes les valeurs décimales ont été arrondies en entier.

Les prochaines étapes du traitement correspondent à des jointures avec d'autres jeux. Nous les détaillerons (*cf. infra* p. 30) une fois les recettes appliquées à l'ensemble des jeux en amont décrites.

3.2.2 Préparation du jeu de Kaggle

Le jeu de Kaggle, nommé *Kaggle* dans Dataiku, correspond à cette partie du flux :

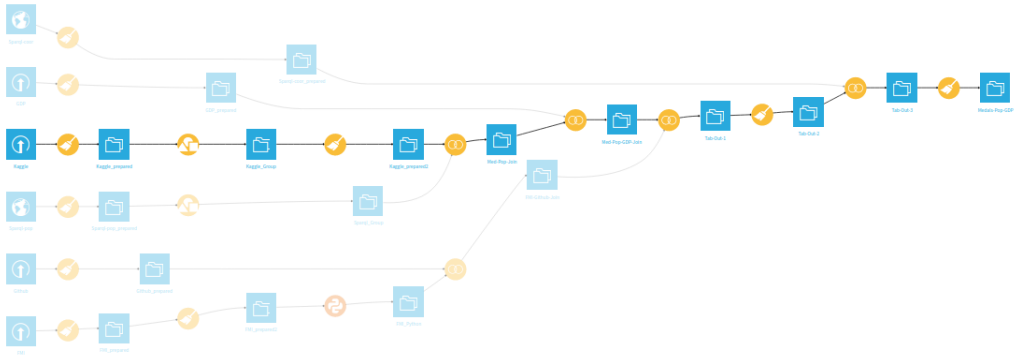


FIGURE 3.2 – Chaîne de traitement du jeu *Kaggle*

En amont des jointures, ce jeu a fait l'objet de trois recettes. La première, une préparation, nous a permis d'uniquement conserver 3 colonnes sur 15 : *NOC*, *Medal*, *Year*. La colonne *Year* a été épurée pour que seules les années des six éditions deux Jeux y apparaissent. Également, les entrées de la colonne *Medal* correspondant à des valeurs nulles (noté « N/A ») ont été supprimées.

L'étape suivante consiste en la réalisation d'un pivot. Les années présentes dans les lignes de la colonne *Year* sont devenues des colonnes dont les lignes ont pris pour valeur les entrées de la colonne *Medal*. 6 nouvelles colonnes ont donc été créés, une par édition des Jeux entre 1996 et 2016.

Enfin, nous avons compté les occurrences des trois types de médailles (bronze, argent, or) et créé des colonnes de sorties pour chaque type pour chaque édition. Par exemple, la colonne *1996* a donné lieu à la création des colonnes *1996 - Bronze* ; *1996 - Silver* ; *1996 - Gold*.

La deuxième recette est un *group*. Le *dataset* présente les médailles reportées par les athlètes, or nous n'avons conservé aucun donnée à leur endroit. Nonobstant, la répartition des médailles repose toujours sur cette logique.

Nous avons donc réalisé un groupement pour réunir les doublons et compter les occurrences de chaque type de médaille remportée pour les six éditions des Jeux. Admettons qu'aux Jeux de 2016, un athlète français ait remporté une médaille de bronze et un autre deux. La recette a pour effet d'afficher le chiffre 3 pour l'entrée « France » sur la colonne *2016 - Bronze*.

La troisième et dernière recette avant la réalisation des jointures est également une préparation. Elle nous a permis, pour chacune des six éditions, de créer une nouvelle colonne présentant la somme de toutes les médailles remportées par les pays. En guise d'exemple, nous savons qu'à l'édition de 2008, l'Australie a remporté un total de 110 médailles.

3.2.3 Préparation du jeu de Wikidata (population)

Le jeu généré grâce à la requête SPARQL sur la population, nommé *Sparql-pop*, correspond à cette partie du flux :

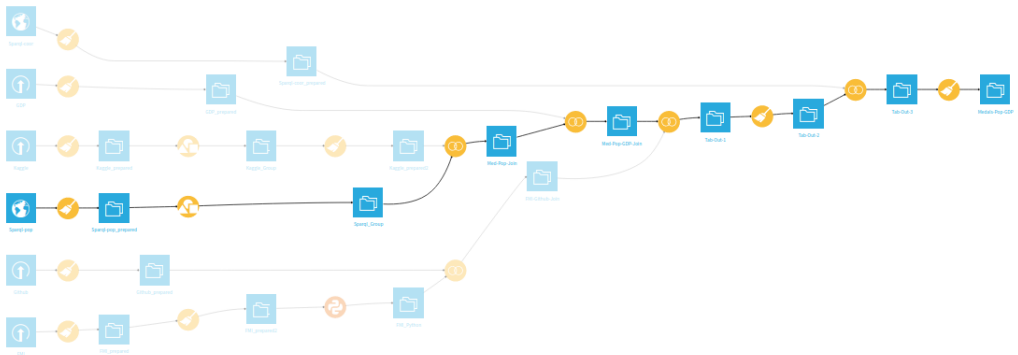


FIGURE 3.3 – Chaîne de traitement du jeu *Sparql-pop*

Ce jeu a été soumis à deux recette. Une préparation où nous avons supprimé les 8 colonnes dont le suffixe était autre que « .value » – lesquelles n'étaient pas sollicitées par notre requête SPARQL (*cf. supra* p. 10). Trois colonnes ont été renommées : *cio.value* est devenu *NOC* ; *isoCode.value* est devenu *Pays* et *paysLabel.value* est devenu *Nom - Pays*.

Nous avons modifié le format des dates : sur l'année, le jour, le mois et l'horaire, nous n'avons conservé que l'année dans une colonne de sortie avant de supprimer la colonne initiale. Les années des six éditions mises à part, toutes les autres ont été supprimées et un nouveau pivot a été réalisé : les années en ligne sont passées en colonne et ont pris pour valeur le nombre d'habitants par pays. Ce pivot a permis la suppression des colonnes *Année* et *population.value*. Enfin, les colonnes des années ont été renommées selon un motif commun : *1996* est devenu *1996 - Pop*.

La dernière étape de cette recette consiste en la suppression de trois valeurs vides présentes dans la colonne *NOC*.

À l'instar du jeu de Kaggle, nous avons appliqué une recette *group* au jeu de Wikidata. Le groupe s'est effectué sur les colonnes *NOC*, *Pays* et *Nom* -

Pays afin de regrouper le nombre d'habitants par année sur une même ligne.

3.2.4 Préparation du jeu de GitHub

Le jeu provenant de la plateforme GitHub, nommé *Github* dans Dataiku, correspond à cette partie du flux :

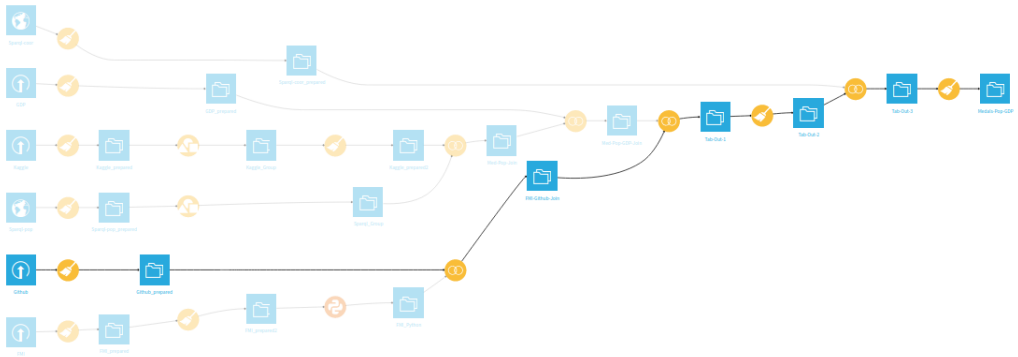


FIGURE 3.4 – Chaîne de traitement du jeu *Github*

Le jeu n'a nécessité qu'une maigre recette de préparation : nous avons supprimé les colonnes *Alpha-2 Code* et *Numeric Code* pour ne conserver que la colonne *Alpha-3 Code* pour faciliter les jointures.

Nous avons également supprimé les entrées que Dataiku ne reconnaissait pas comme pays. Enfin, nous avons renommé 2 colonnes : *Country* est devenu *Pays* et *Alpha-3 Code* est devenu *Code*.

3.2.5 Préparation du jeu du FMI

Le jeu du Fonds Monétaire International, nommé *FMI* dans Dataiku, correspond à cette partie du flux :

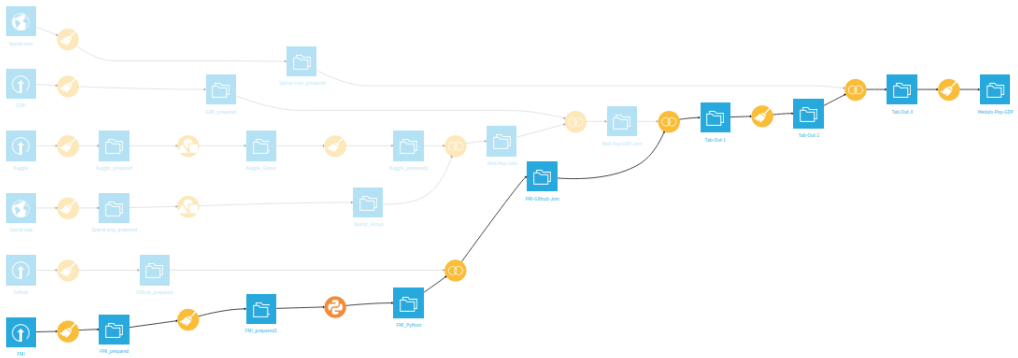


FIGURE 3.5 – Chaîne de traitement du jeu *FMI*

À la faveur de la première recette – une préparation – nous avons supprimé de la colonne *Unit Name* les lignes où la valeur était « Domestic currency » pour ne conserver que le pourcentage du Produit Intérieur Brut (« Percent of GDP ») investi par les pays dans le domaine du sport.

Nous avons uniquement conservé 7 colonnes : *Country Name* ainsi que les dates des six éditions des Jeux. La dernière étape de la préparation consistait en la suppression de lignes vides pour la colonne *2016*. À partir de ce moment, Dataiku se figeait et nous laissait dans l'impossibilité d'ajouter de nouvelles étapes. Nous avons donc dû ajouter une autre recette de préparation pour que le *run* fonctionne et reprendre le traitement.

Dans cette deuxième préparation, les colonnes des années des Jeux ont été renommées selon le motif suivant : *1996* devient *1996 - FMI*.

Il nous a fallu manuellement modifier 13 valeurs dans la colonne *Country Name* pour une meilleure reconnaissance des pays par Dataiku. Par exemple : *Russian Federation*, *Czech Rep.* ne sont pas reconnus, tandis que *Russia* et *Czech Republic* le sont. Ces modifications ont donné lieu à la création d'une colonne de sortie *Pays* car il nous était impossible de modifier les noms

des pays directement dans la colonne d'origine. Nous avons naturellement supprimé la colonne *Country Name*, une fois ces modifications effectuées.

Nous avons rencontré un problème de données aberrantes dans les valeurs indiquées pour les investissements par rapport au PIB. La nature du problème repose sur des conversions mathématiques fautives : les pour mille et pour cent ont subi la même opération. Nous avons donc entrepris de corriger ces données par l'intermédiaire d'une recette Python que voici :

```
import dataiku
import pandas as pd
from dataiku import pandasutils as pdu

FMI_prepared2 = dataiku.Dataset("FMI_prepared2")
FMI_prepared2_df = FMI_prepared2.get_dataframe()
colonnes_a_traiter = ['1996 - FMI', '2000 - FMI', '2004 - FMI',
    '2008 - FMI', '2012 - FMI', '2016 - FMI']

for colonne in colonnes_a_traiter:
    condition = FMI_prepared2_df[colonne] > 0.1
    FMI_prepared2_df.loc[condition, colonne] /= 10

FMI_prepared2_df.drop_duplicates(subset=['Pays'], keep='first',
    inplace=True)
FMI_Python = dataiku.Dataset("FMI_Python")
FMI_Python.write_with_schema(FMI_prepared2_df)
```

Le code ci-dessus intervient sur les colonnes des six éditions des Jeux :

```
colonnes_a_traiter = ['1996 - FMI', '2000 - FMI', '2004 - FMI',
    '2008 - FMI', '2012 - FMI', '2016 - FMI']
```

Le contenu de ces colonnes est modifié grâce à une boucle `for` : si elles contiennent des valeurs supérieures à `0.1`, ces dernières sont divisées par `10` pour obtenir la juste valeur d'investissement :

```
for colonne in colonnes_a_traiter:
    condition = FMI_prepared2_df[colonne] > 0.1
    FMI_prepared2_df.loc[condition, colonne] /= 10
```

La bonne valeur des données est ainsi rétablie.

3.2.6 Préparation du jeu de Wikidata (coordonnées)

Rappelons que nous avons dû trouver des données à même de corriger les coordonnées impropres du jeu de GitHub. Ce jeu, également issu d'une requête SPARQL et nommé *Sparql-coor* dans Dataiku, correspond à cette partie du flux :

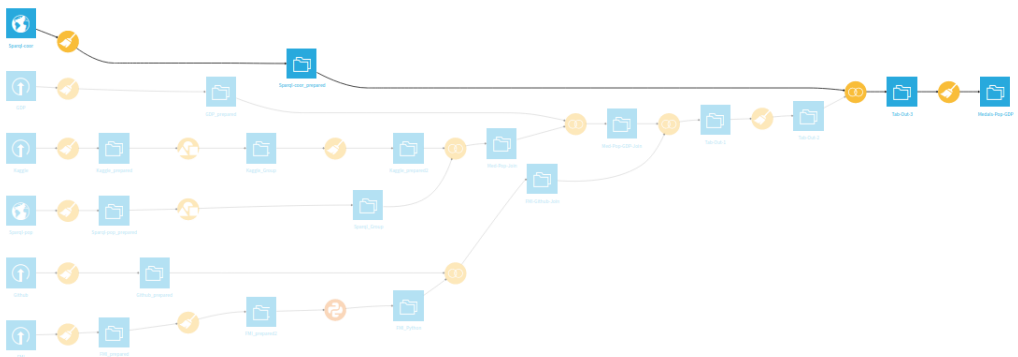


FIGURE 3.6 – Chaîne de traitement du jeu *Sparql-coor*

Une seule recette de préparation a été derechef nécessaire en amont des jointures. Celle-ci ne conserve que les colonnes *noc.value* et *geopoint.value*.

Les deux autres étapes de la recette nous permettent de correctement typer et extraire les coordonnées de la colonne *geopoint.value* afin de correctement les exploiter pour réaliser nos datavisualisations. Les données sont écrites comme suit : « point(11.0 65.0) ». Nous avons pu les extraire et créer une colonne *Latitude* et une autre *Longitude*.

Nonobstant, les données ainsi typées sont effectivement reconnues par le logiciel de datavisualisation Tableau Public comme étant des coordonnées mais équivalent toutes à « Null » une fois importées. Nous avons essayé de résoudre le problème directement depuis Tableau Public, sans succès car le logiciel considérait les coordonnées comme des chaînes de caractères et ne pouvait bien entendu pas calculer une longitude et une latitude.

Nous avons finalement compris que le problème était la casse : pour que les données soient correctement typées, il fallait que tous les caractères soient en haut-de-casse (non pas « point(11.0 65.0) » mais « POINT(11.0 65.0) »). Nous avons donc ajouté cette étape dans la recette avant d'extraire les longitudes et latitudes dans 2 colonnes distinctes.

3.2.7 Première jointure

La première jointure concerne les jeux préparés *Kaggle* et *Sparql-pop*, selon la colonne *NOC*. Il en résulte la création du jeu *Med-Pop-Join*, lequel contient 33 colonnes. 2 colonnes sur le code des pays, 1 sur leur nom, 6 colonnes sur la population des années correspondant aux éditions des Jeux et 4 colonnes par année : 3 pour les types de médailles et 1 pour la somme de ces dernières.

Cette *left join* s'est réalisée sans suppression et n'a pas occasionné de pertes significatives de données.

3.2.8 Deuxième jointure

À la suite de cette première jointure s'effectue une deuxième *left join* : les jeux *Med-Pop-Join* et *GDP* sont ainsi regroupés – également selon la colonne *NOC* – engendrant le jeu *Med-Pop-GDP-Join*. 6 colonnes ont été alors ajoutées aux 33 du jeu *Med-Pop-Join*. Elles correspondent au PIB des pays pour chaque année d'édition des Jeux Olympiques.

3.2.9 Troisième jointure

Une troisième *left join* croise les jeux *FMI* et *Github*, selon la colonne *Pays* du jeu *FMI*. Le jeu *FMI-Github-Join* est créé et contient 10 colonnes. Les données du Fonds Monétaire International ont ainsi été enrichies par le code et les coordonnées des pays, fournis par le jeu de GitHub.

3.2.10 Quatrième jointure

La quatrième jointure rassemble les jeux *Med-Pop-GDP-Join* et *FMI-Github-Join*. Dans la mesure où les médailles remportées constituent les données les plus importantes à nos yeux, la *left join* a été réalisée de telle sorte à ce que les pertes, s'il devait y en avoir, aient des conséquences sur les données du jeu *FMI-Github-Join* – non pas sur celles du jeu *Med-Pop-GDP-Join*.

3.2.11 Préparation du jeu final

Le jeu créé, nommé *Tab-Out-1* a subi une recette de préparation afin de nettoyer (nous le pensions) une dernière fois les données. 42 de ses 47 colonnes ont été réorganisées selon un même schéma pour chaque édition des Jeux :

FIGURE 3.7 – Extrait du jeu *Tab-Out-1*

Les 5 autres colonnes présentent le nom des pays, leurs codes et leurs coordonnées. Nous pensions que ce jeu ainsi nettoyé, nommé *Tab-Out-2* était la dernière étape du flux. Cependant, une fois importé dans Tableau Public, les coordonnées se sont révélées inutilisables. C'est pourquoi la seconde requête SPARQL sur les coordonnées des pays était nécessaire.

Nous avons donc réalisé une dernière jointure sur les jeux *Sparql-coor* et *Tab-Out-2*. Il s'agit d'une *inner join* selon le code NOC. Le jeu alors créé, *Tab-Out-3* voit ses valeurs de coordonnées corrigées.

Ce jeu de sortie a fait l'objet d'un dernier nettoyage tardif. D'une part, la colonne *2004 - GDP* avait échappé à l'étape d'organisation précédemment décrite (*cf. supra* p. 32). D'autre part, nous ne suivions pas un mode de nommage de colonnes tout à fait rigoureux. Comme nous avions prévu de présenter des données en langue anglaise, nous avons renommé toutes les colonnes dont tout ou partie de l'intitulé était encore en français.

Est créé notre jeu final : *Medals-Pop-GDP*, au terme du flux ci-dessous.

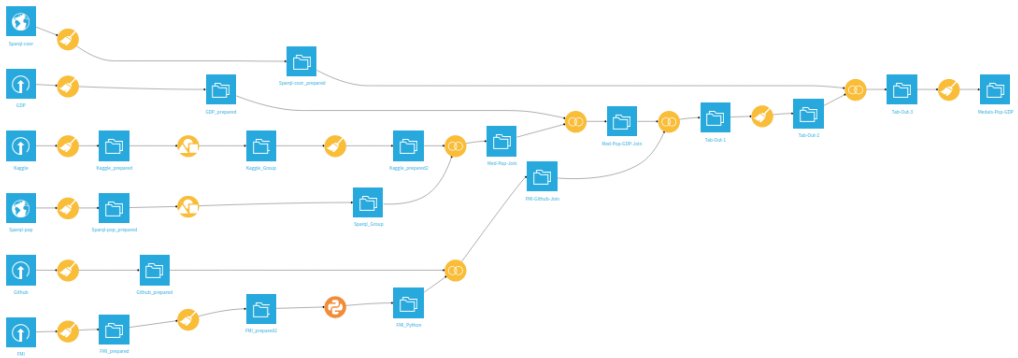


FIGURE 3.8 – Flux de traitement sur les médailles et les investissements

3.3 Chaîne de traitement du second flux

Notons que ce deuxième flux est composé de trois petits flux qu'il ne fait pas sens de joindre au bout de chaîne sur Dataiku. La mise en relation des quatre jeux de données finaux sera l'objet des datavisualisation.

Le premier mini-flux a pour sujet les infrastructures relatives à la natation du Royaume-Uni, le deuxième mini-flux *idem* à l'égard de la France et le troisième concerne les médailles remportées par les deux pays.

3.3.1 GBR : préparation des jeux de *Sport England*

Les jeux *Swimming-Pools* et *Geographic* ont tous deux été préparés en amont de leur jointure. Ils correspondent à cette partie du flux général :

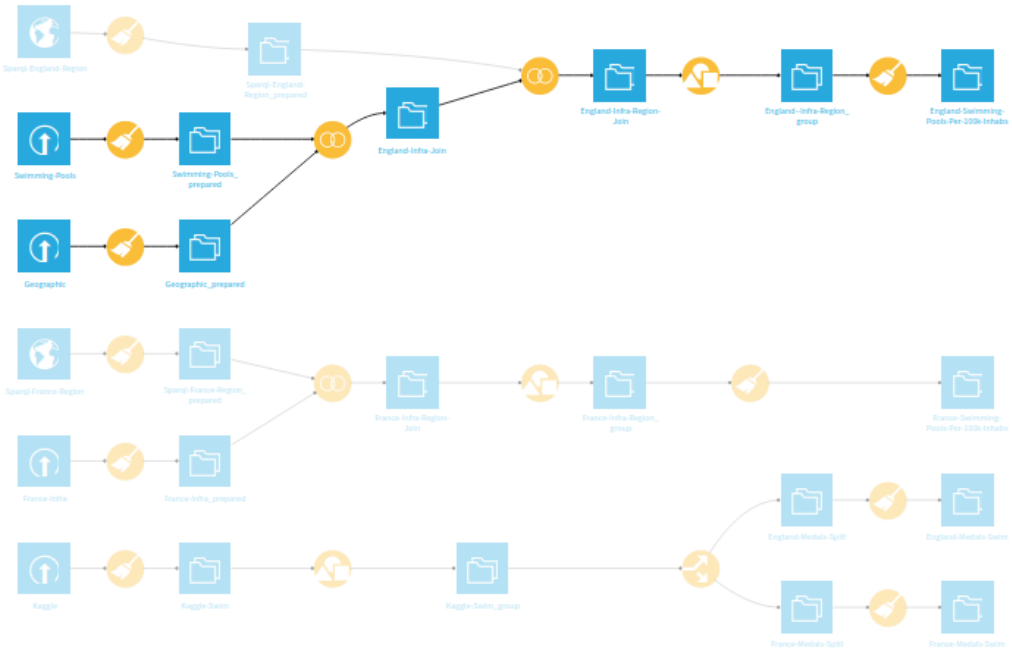


FIGURE 3.9 – Chaîne de traitement des jeux *Swimming-Pools* et *Geographic*

Du jeu *Swimming-Pools* n'a été gardée que la colonne *FacilityID* – précisément pour réaliser une jointure avec le jeu *Geographic*. Dans la mesure où chaque identifiant unique d'infrastructure compte pour 1, nous n'avons besoin d'aucune autre colonne pour la suite du traitement.

Un processus similaire de sélection a été appliqué au jeu *Geographic* : nous n'avons conservé que 4 colonnes, *FACILITYID*, *Region Name*, *Latitude*, *Longitude*. La présence des identifiants uniques dans les deux jeux nous a permis de réaliser une jointure interne et ainsi de lier les infrastructures de natation aux régions d'Angleterre, dans le jeu *England-Infra-Join* nouvellement créé.

3.3.2 GBR : préparation du jeu de Wikidata

La requête SPARQL, ici nommée *Sparql-England-Region*, correspond à cette partie du flux :

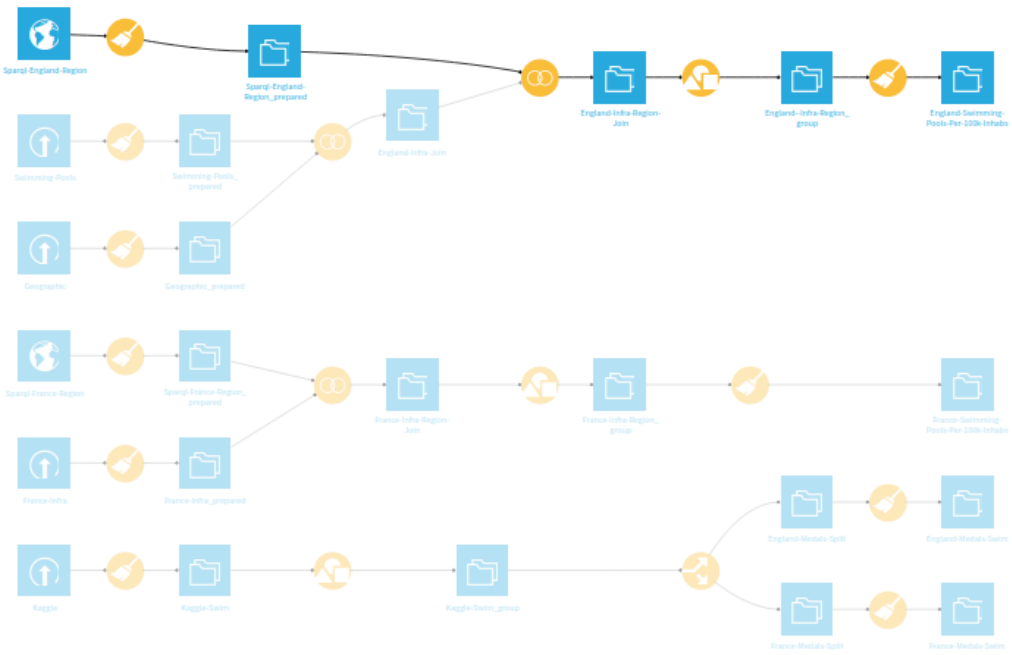


FIGURE 3.10 – Chaîne de traitement du jeu *Sparql-England-Region*

Avant de réaliser la jointure avec les deux autres jeux de ce mini-flux, nous appliquons une recette de préparation à celui-ci. Des colonnes initiales, nous n'en conservons que 3 : *population.value*, *geopoint.value* et *RegionLabel.value* – soit le nombre d'habitants des régions, leurs coordonnées et leur nom.

Nous évacuons les lignes vides de la colonne *population.value* et renommons manuellement certaines données de la colonne *RegionLabel.value* pour une meilleure reconnaissance, à la fois par Dataiku et Tableau Public. Pour

que les coordonnées soient reconnues et exploitables (*cf. supra* p. 30), nous convertissons toutes les valeurs textuelles de la colonne *geopoint.value* en haut-de-casse. Enfin, nous extrayons les valeurs de longitude et de latitude de cette même colonne et créons ainsi les colonnes *Latitude* et *Longitude*.

3.3.3 GBR : jointure et préparation du jeu final

La jointure interne des jeux *Sparql-England-Region* et *England-Infra-Join* a été effectuée sur le noms des régions. Il en ressort le jeu *England-Infra-Region-Join*, composé de 5 colonnes – le nom, les coordonnées, la population des régions et l'identifiant des infrastructures.

Nous avons appliqué une recette de groupement sur ce jeu, selon les colonnes *population.value* et *FacilityID* pour n'avoir qu'une ligne par région et par nombre d'habitants. De ce fait, la colonne *FacilityID* a donné lieu à la nouvelle colonne *count*, laquelle présente la somme des infrastructures par région. Cette donnée est au cœur de la dernière recette. Celle-ci nous permet, par l'intermédiaire d'un produit en croix, de compter le nombre d'infrastructure par région pour 100.000 habitants et de l'afficher dans la colonne *Swimming pools per 100k inhabitants*. Notre jeu final, *England-Swimming-Pools-Per-100k-Inhabs* est ainsi créé.

3.3.4 FRA : préparation du jeu du ministère

Le jeu du ministère (*France-Infra*), à l'instar du jeu de Wikidata sur les régions françaises, a été préparé avant la jointure nécessaire. Il correspond à cette partie du flux :

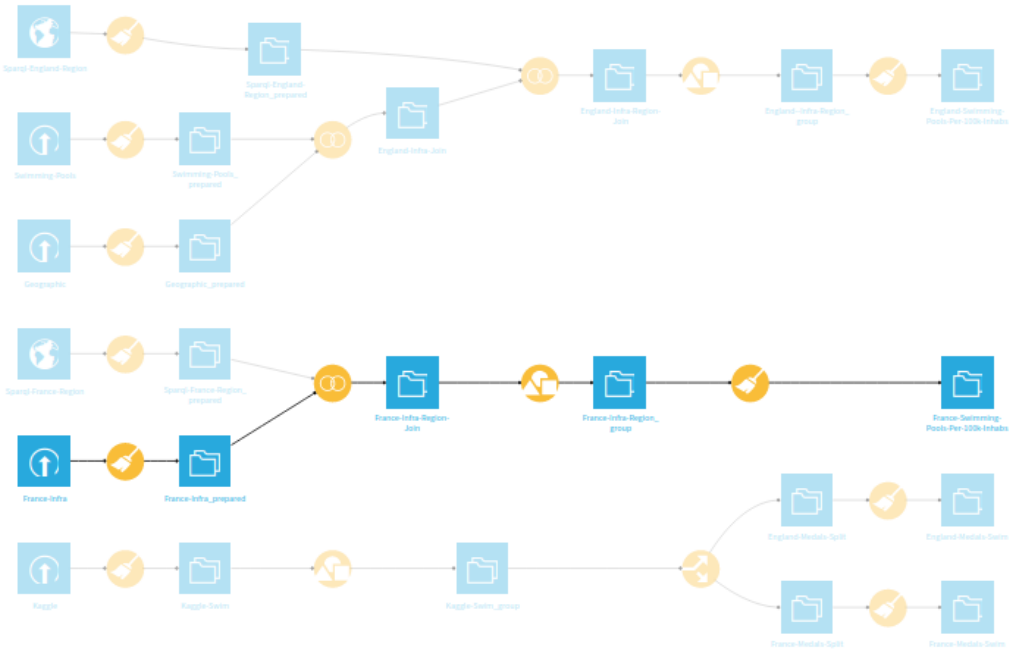


FIGURE 3.11 – Chaîne de traitement du jeu *France-Infra*

Seules 4 colonnes ont été conservées : *Type d'équipement sportif*, *Longitude (WGS84)*, *Latitude (WGS84)* et *Région Nom*. Nous avons ensuite examiné plus en détails les lignes de la colonne *Type d'équipement sportif* et avons pu établir une liste de 5 valeurs relatives à la natation. Nous avons supprimé toutes les lignes qui n'y correspondaient pas. Ces valeurs ont été communément renommées « picisines_centres_aquatiques ».

3.3.5 FRA : préparation du jeu de Wikidata

Le jeu de Wikidata sur les régions de France, nommé *Sparql-Region-France* correspond à cette partie du flux :

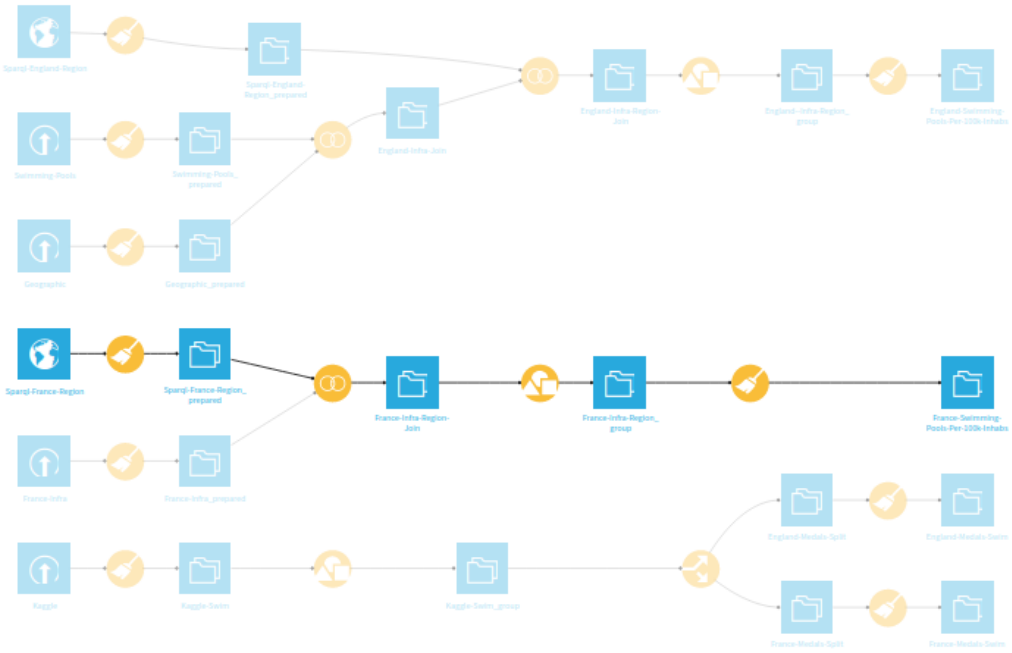


FIGURE 3.12 – Chaîne de traitement du jeu *Sparql-Region-France*

Seules les colonnes *population.value*, *geopoint.value* et *RegionLabel.value* ont été conservées et renommées. De la même manière que pour le jeu Wikidata sur les régions d'Angleterre, les lignes de la colonne *geopoint.value* ont été mises en haut-de-casse, les longitudes et latitudes extraites dans deux nouvelles colonnes, *Longitude*, *Latitude*.

3.3.6 FRA : jointure et préparation du jeu final

La jointure interne des jeux *France-Infra* et *Sparql-France-Region* a été réalisée selon la colonne des noms de région. Le jeu ainsi créé, *France-Infra-Region-Join* est tout à fait similaire au jeu *England-Infra-Region-Join*. Les dernières recettes sont *de facto* analogues.

Nous réalisons un groupement sur les types d'équipement et effectuons derechef un produit en croix pour générer une colonne *Swimming pools per 100k inhabitants*. Le dernier jeu de ce mini-flux, nommé *France-Swimming-Pools-Per-100k-Inhabitants* émerge alors.

3.3.7 Médailles : préparation du jeu de Kaggle

Enfin, le jeu *Kaggle* issu de notre premier flux (précisément du jeu nommé *Kaggle_prepared2*) correspond à cette partie de la chaîne :

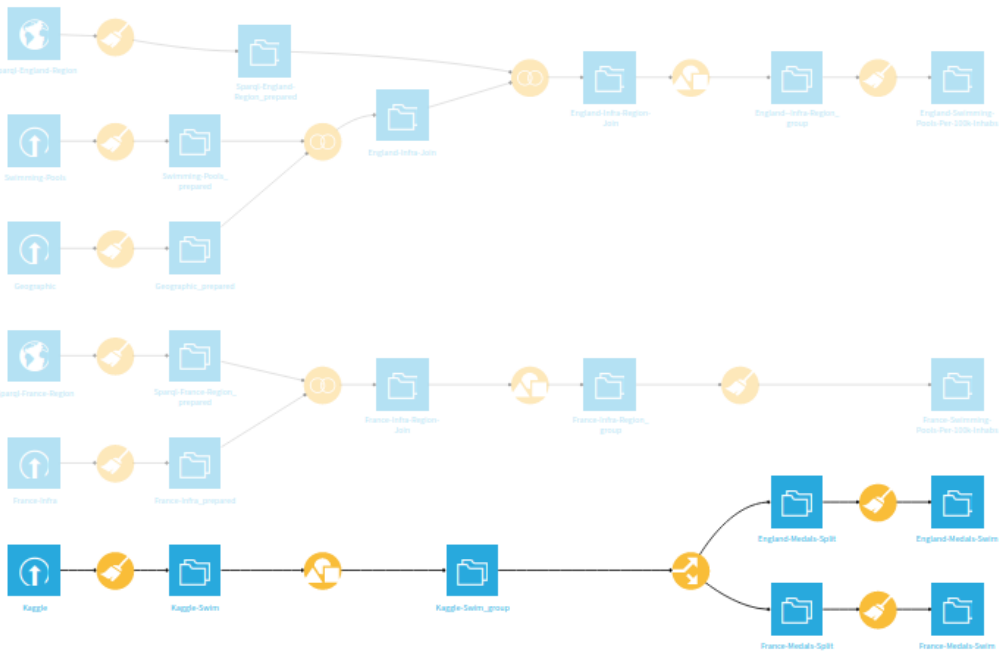


FIGURE 3.13 – Chaîne de traitement du jeu *Kaggle*

Dans une première recette de préparation, nous avons identifié les lignes de la colonne *Sport* relatives à la natation, les avons renommées « Swim-

ming_sports » et avons supprimé toutes les autres. Également, nous n'avons conservé que les lignes où le code NOC est équivalent à « FRA » et « GBR ».

Nous avons ensuite appliqué une recette de groupement pour que seules 2 lignes affichent l'ensemble des médailles remportées, d'une part par la France, d'autre part par le Royaume-Uni. L'avant-dernière recette sépare le jeu groupé en deux jeux distincts : l'un pour les médailles remportées par la France, l'autre pour les médailles remportées par le Royaume-Uni.

3.3.8 Médailles : préparation du jeu final

Les deux jeux alors créés, *England-Medals-Split* et *France-Medals-Split* ont été identiquement nettoyés : la langue et la casse des noms de colonnes harmonisées et leur ordre basé sur le même motif que celui adopté pour le jeu du premier flux (*cf. supra* p. 32). Nos deux jeux finaux, *England-Medals-Swim* et *France-Medals-Swim* closent la chaîne de traitement de ce mini-flux.

Notre deuxième grand flux a donc abouti à la création de quatre jeux de données, lesquels seront tous exploités et liés dans nos datavisualisations :

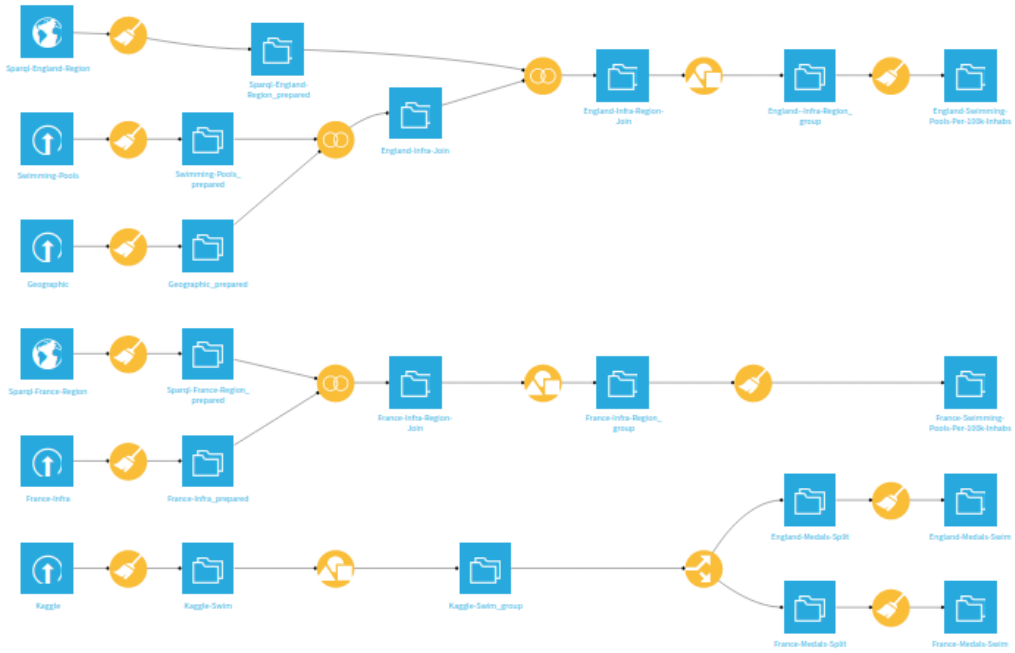


FIGURE 3.14 – Flux de traitement sur les médailles de natation

Visualisation des données

4.0.1 Médailles remportées par pays

4.0.2 Infrastructures de natation : France

4.0.3 Infrastructures de natation : Royaume-Uni

Il nous a été impossible de créer cette visualisation directement à partir de nos jeux finaux générés sur Dataiku. Utiliser les coordonnées ne nous permet d'afficher que des points, or nous chercherons à discerner les régions grâce à leur contour. Nous pensions donc que le nom des régions nous le permettrait mais leur utilisation provoque un conflit dans Tableau Public.

En revanche, cela fonctionne avec l'équivalent des codes NOC des régions. Nous avons alors songé à adapter notre requête SPARQL relative (*cf. supra* p. 17) pour récupérer ces données. Cependant, une telle propriété n'existe pas dans le service de requête de Wikidata. Nous avons donc dû l'ajouter manuellement directement au sein de Tableau Public.

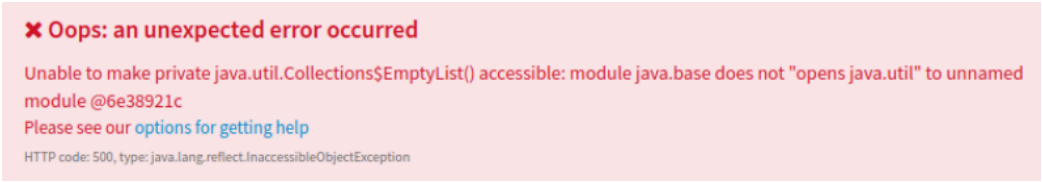
Remarques et péripéties

5.1 Typage de données

Un grand défi du traitement a été le typage de données, autant pour Dataiku que pour Tableau Public. Nous découvrions ces deux logiciels et parvenir à les faire dialoguer n'a pas été sans peine (exemplairement à propos des coordonnées, *cf. supra* p. 30). Dataiku retypait des données automatiquement en dépit de nos modifications manuelles et chaque exécution de recette et réalisation de datavisualisation était mue par le risque d'un bogue de typage et l'obligation d'intervenir derechef dans les flux.

5.2 Bogue Dataiku

La semaine du 22 janvier a été le lieu d'une mauvaise surprise. Alors que nous voulions examiner le premier flux sur Dataiku pour rédiger le présent journal de bord, il nous a été subitement impossible de visualiser les jeux. L'interface générale fonctionnait, nous pouvions voir le flux, cliquer sur les icônes de recettes ou de jeux mais ce faisant, la partie de l'écran devant afficher les données était vide. L'erreur renvoyée est la suivante :



Il est probable que cette erreur ait été créée après l'installation d'une version de *Java Development Kit* (à l'occasion d'un cours de XQuery) supérieure à celle prise en charge par Dataiku, sur la machine où a été réalisé l'ensemble du traitement des données. Il aurait fallu rétrograder la version de JDK mais nous craignions que celle manipulation n'entraîne des difficultés sur l'ensemble des projets dépendants de JDK.

Nous avons été forcés de nous appuyer sur d'autres ordinateurs pour consulter les flux et terminer le traitement (la réalisation du deuxième flux n'avait alors pas commencé, et celle du premier n'était pas tout à fait achevée). Cela a causé d'importants soucis organisationnels en cette fin de mois.

5.3 Sur l'utilité de Wikidata

Au cours du traitement Wikidata s'est révélé être un outil formidable pour l'enrichissement et le traitement des données. Le système de construction des requêtes SPARQL nécessite une certaine prise en main mais nous permet de très larges interrogations de la base de données.

Les jeux générés sont d'une qualité exemplaire et sont d'un grand intérêt pour correctement réaliser des jointures à l'aide des nombreux codes, labels et identifiants récupérables.

Sitographie

120 years of Olympic history : athletes and results. URL : https://www.kaggle.com/datasets/heesoo37/120-years-of-olympic-history-athletes-and-results (visité le 17/01/2024).

Active Places Power. URL : https://www.activeplacespower.com/ (visité le 20/01/2024).

Copil, Alexandru-Paul. *Countries codes and coordinates*. URL : https://gist.github.com/cpl/3dc2d19137588d9ae202d67233715478 (visité le 25/01/2024).

Data ES - Base de données. URL : https://equipements.sports.gouv.fr/explore/dataset/data-es/information/ (visité le 17/01/2024).

Download entire World Economic Outlook database, April 2019. IMF. URL : https://www.imf.org/en/Publications/WEO/weo-database/2023/October/download-entire-database (visité le 17/01/2024).

Point in time. URL : https://www.wikidata.org/wiki/Property:P585 (visité le 20/01/2024).

Population. URL : https://www.wikidata.org/wiki/Property:P1082 (visité le 20/01/2024).

Wikidata prefixes (E49) - Wikidata. URL : https://www.wikidata.org/wiki/EntitySchema:E49 (visité le 20/01/2024).

Wikidata Query Service. URL : https://w.wiki/8uTN (visité le 27/01/2024).

Wikidata Query Service. URL : https://w.wiki/8yne (visité le 28/01/2024).

Wikidata Query Service. URL : https://w.wiki/8uHH (visité le 20/01/2024).

Wikidata Query Service. URL : https://w.wiki/8yqy (visité le 28/01/2024).

Wikidata Query Service. URL : https://w.wiki/8ynm (visité le 28/01/2024).

World Bank Open Data. World Bank Open Data. URL : https://data.worldbank.org (visité le 20/01/2024).

Table des matières