Groupe de recherche interdisciplinaire en informatique de la santé

Documentation SDC - Modèles de Conception de Requêtes (MCR)

OntoRelQuery

MCR

Mohamed Amin GAIED *(Mohamed.Amin.Gaied@USherbrooke.ca)*

—

MAD/«depot»/MCR_SDC, version 1.0.0.a, en date du 2025-06-18

1. Introduction aux MCR

Sommaire

Cette documentation fonctionnelle (SDC) explique comment utiliser les MCR avec des exemples concrets. Pour la documentation technique détaillant l'architecture et l'implémentation, consultez la Documentation technique des MCR (SES).

1.1. Qu'est-ce qu'un MCR ?

Les MCR agissent comme une interface entre le monde ontologique (classes, propriétés, relations) et le monde relationnel (tables, colonnes, jointures), permettant aux utilisateurs de formuler des requêtes en termes ontologiques sans avoir à connaître les détails du schéma de base de données sous-jacent. Ils encapsulent les patrons de conception de requêtes fréquemment utilisés et les rendent facilement accessibles via une interface en ligne de commande.

1.2. Pourquoi utiliser les MCR ?

L'utilisation des MCR présente plusieurs avantages :

- **Abstraction** : Les MCR permettent de formuler des requêtes en termes ontologiques, plus proches du domaine métier
- **Réutilisation** : Les modèles de requêtes fréquemment utilisés peuvent être réutilisés facilement
- **Standardisation** : Les MCR offrent une approche standardisée pour la construction de requêtes
- **Simplicité** : Les utilisateurs n'ont pas besoin de connaître les détails du schéma de base de données
- **Flexibilité** : Les MCR peuvent être combinés pour former des requêtes complexes
- **Maintenabilité** : Les modifications du schéma de base de données n'affectent pas les requêtes ontologiques
- **Évolutivité** : De nouveaux modèles peuvent être ajoutés facilement pour répondre à de nouveaux besoins

1.3. Cas d'utilisation typiques

Les MCR sont particulièrement utiles dans les scénarios suivants :

- **Exploration de données** : Recherche d'instances de classes ontologiques spécifiques
- **Analyse de relations** : Exploration des relations entre différentes entités
- **Requêtes complexes** : Construction de requêtes impliquant plusieurs classes et propriétés

- **Agrégation de données** : Calcul de statistiques sur des ensembles de données
- **Intégration de données** : Combinaison de données provenant de différentes sources
- **Recherche de chemins** : Identification des chemins entre concepts ontologiques
- **Validation de données** : Vérification de la cohérence des données par rapport à l'ontologie
- **Génération de rapports** : Création de rapports basés sur des modèles prédéfinis

2. Guide d'utilisation des MCR

2.1. Comment lister les modèles disponibles

Pour voir la liste des modèles de requêtes disponibles, utilisez la commande `mcr-list` dans l'interface en ligne de commande :

```
> mcr-list
```

Cette commande affiche tous les modèles disponibles, regroupés par catégorie, avec leur description, leurs options requises et leur statut de disponibilité.

Exemple de sortie :

```
╔═══════════════════════════════════════════════════════════════════════════════╗
║                    AVAILABLE QUERY MODELS                                     ║
╚═══════════════════════════════════════════════════════════════════════════════╝

Use the execute --type MODEL_NAME command or !MODEL_NAME shortcut to execute a model.
Example: execute --type MS_SC or !MS_SC

[Simple Models] - Class selection with various constraints
1. MS_SC - Class selection [Available]
   Required options: class
   Usage: !MS_SC or execute --type MS_SC

2. MS_SCPO - Class selection by object property [Available]
   Required options: class, objectProperty
   Usage: !MS_SCPO or execute --type MS_SCPO

[Iterative Models] - Path-based queries between ontological concepts
1. MI_AP - Querying all possible paths between two classes [Available]
   Required options: from, to
   Usage: !MI_AP or execute --type MI_AP

[Combinatorial Models] - Combination of multiple query types
1. MC_COMBINATORIAL - Combinatorial model [Available]
   Required options: subModels
   Usage: !MC_COMBINATORIAL or execute --type MC_COMBINATORIAL
```

2.2. Comment exécuter un modèle

Il existe deux façons d'exécuter un modèle de requête :

1. Utiliser la commande `execute` avec l'option `--type` :

```
> execute --type MS_SC
```

1. Utiliser le raccourci `!` suivi du nom du modèle :

```
> !MS_SC
```

Dans les deux cas, le système vous demandera de fournir les paramètres requis pour le modèle sélectionné.

2.3. Comment interpréter les résultats

Les résultats des requêtes sont affichés dans la console sous forme de tableau et sont également exportés dans des fichiers pour une analyse ultérieure.

Exemple de résultat pour une requête MC_CI :

```
╔═══════════════════════════════════════════════════════════════════════════╗
║                        QUERY RESULTS                                      ║
╚═══════════════════════════════════════════════════════════════════════════╝

Table: patient
╔═══════════════╦═══════════════╦═══════════════╦═══════════════╗
║ patient_id    ║ first_name    ║ last_name     ║ birth_date    ║
╠═══════════════╬═══════════════╬═══════════════╬═══════════════╣
║ 1             ║ John          ║ Doe           ║ 1980-01-15    ║
║ 2             ║ Jane          ║ Smith         ║ 1975-05-20    ║
║ 3             ║ Robert        ║ Johnson       ║ 1990-11-08    ║
╚═══════════════╩═══════════════╩═══════════════╩═══════════════╝

Results exported to: /path/to/results/2025-06-18_08-30-45/
```

Les résultats exportés comprennent :

- **first_query.sql** : SQL pour la requête de catalogue
- **first_result.txt** : Résultats de la requête de catalogue
- **second_query.sql** : SQL pour la requête principale
- **second_result.txt** : Résultats de la requête principale

3. Exemples d’utilisation

3.1. Modèles simples (MS_*)

Les modèles simples permettent de sélectionner des instances de classes ontologiques avec diverses contraintes.

3.1.1. MS_SC : Sélection simple de classe

Ce modèle permet de sélectionner toutes les instances d’une classe ontologique.

Paramètres requis : * `class` : Nom de la classe ontologique à sélectionner

Exemple :

```
> !MS_SC
Enter value for class: http://purl.obolibrary.org/obo/HOSO_0000060
```

Résultat :

```
Table: HOSO_0000060
```

```
╔═══════════════════╦═══════════════════════════════════════════════╗
║ HOSO_0000060_uid  ║ public provincial health insurance record     ║
╠═══════════════════╬═══════════════════════════════════════════════╣
║ 1                 ║ Record 001                                    ║
║ 2                 ║ Record 002                                    ║
║ 3                 ║ Record 003                                    ║
╚═══════════════════╩═══════════════════════════════════════════════╝
```

3.1.2. MS_SCPO : Sélection de classe par propriété d'objet

Ce modèle permet de sélectionner des instances d'une classe ontologique liées à d'autres instances par une propriété d'objet.

Paramètres requis : * `class` : Nom de la classe ontologique à sélectionner * `objectProperty` : Nom de la propriété d'objet à utiliser

Exemple :

```
> !MS_SCPO
Enter value for class: http://purl.obolibrary.org/obo/HOSO_0000060
Enter value for objectProperty: http://purl.obolibrary.org/obo/IAO_0000136
```

Résultat :

```
Table: HOSO_0000060_IAO_0000136_HOSO_0000062
╔═══════════════════╦═══════════════════════════════════════════════╦═══════════════════════════════════════════╗
║ HOSO_0000060_uid  ║ public provincial health insurance record     ║ primary human health insurance beneficiary ║
╠═══════════════════╬═══════════════════════════════════════════════╬═══════════════════════════════════════════╣
║ 1                 ║ Record 001                                    ║ Beneficiary 001                           ║
╠═══════════════════╬═══════════════════════════════════════════════╬═══════════════════════════════════════════╣
║ 2                 ║ Record 002                                    ║ Beneficiary 002                           ║
╠═══════════════════╬═══════════════════════════════════════════════╬═══════════════════════════════════════════╣
║ 3                 ║ Record 003                                    ║ Beneficiary 003                           ║
╚═══════════════════╩═══════════════════════════════════════════════╩═══════════════════════════════════════════╝
```

3.1.3. MS_SCPD : Sélection de classe par propriété de données

Ce modèle permet de sélectionner des instances d'une classe ontologique avec leurs propriétés de données.

Paramètres requis : * `class` : Nom de la classe ontologique à sélectionner * `dataProperty` : Nom de la propriété de données à utiliser

Exemple :

```
> !MS_SCPD
Enter value for class: http://purl.obolibrary.org/obo/HOSO_0000019
Enter value for dataProperty: http://purl.obolibrary.org/obo/OpenLHS-Core_0000059
```

Résultat :

```
Table: HOSO_0000019_OpenLHS-Core_0000059
╔═══════════════════════╦═══════════════════════════════╦═══════════════════╦═══════════╗
║ HOSO_0000019_uid      ║ public provincial health id   ║ has value         ║ @value    ║
╠═══════════════════════╬═══════════════════════════════╬═══════════════════╬═══════════╣
║ 1                     ║ ID001                         ║ value_1           ║ 12345     ║
╠═══════════════════════╬═══════════════════════════════╬═══════════════════╬═══════════╣
║ 2                     ║ ID002                         ║ value_2           ║ 67890     ║
╠═══════════════════════╬═══════════════════════════════╬═══════════════════╬═══════════╣
║ 3                     ║ ID003                         ║ value_3           ║ 24680     ║
╚═══════════════════════╩═══════════════════════════════╩═══════════════════╩═══════════╝
```

3.1.4. MS_SCH : Sélection de classe par hiérarchie

Ce modèle permet de sélectionner des instances d'une classe ontologique en tenant compte des relations hiérarchiques.

Paramètres requis : * `class` : Nom de la classe ontologique à sélectionner * `hierarchy` : Type de relation hiérarchique (subClassOf, superClassOf)

Exemple :

```
> !MS_SCH
Enter value for class: http://purl.obolibrary.org/obo/OpenLHS-Core_0000065
Enter value for hierarchy: subClassOf
Enter value for level: 2
Enter value for traversal: C
```

Résultat :

```
Table: temporal_information_hierarchy
╔═══════════════════════════════╦═══════════════════════════════╦═══════════════════════════════╗
║ temporal information          ║ human birth temporal information ║ HDRN human birth temporal information ║
╠═══════════════════════════════╬═══════════════════════════════╬═══════════════════════════════╣
║ temp_info_1                   ║ birth_info_1                  ║ hdrn_birth_1                  ║
╠═══════════════════════════════╬═══════════════════════════════╬═══════════════════════════════╣
║ temp_info_2                   ║ birth_info_2                  ║ hdrn_birth_2                  ║
╠═══════════════════════════════╬═══════════════════════════════╬═══════════════════════════════╣
║ temp_info_3                   ║ birth_info_3                  ║ hdrn_birth_3                  ║
╚═══════════════════════════════╩═══════════════════════════════╩═══════════════════════════════╝
```

3.1.5. MS_SCA : Sélection de classe avec agrégation

Ce modèle permet de sélectionner des instances d'une classe ontologique et d'appliquer des fonctions d'agrégation.

Paramètres requis : * `class` : Nom de la classe ontologique à sélectionner * `aggregationColumn` : Colonne à agréger (collectée interactivement) * `aggregationFunction` : Fonction d'agrégation à appliquer (collectée interactivement)

Exemple :

```
> !MS_SCA
Enter value for class: http://purl.obolibrary.org/obo/PDRO_0000042

Available columns for aggregation:
1. PDRO_0000042_uid
2. record_id
3. date
0. * (All columns)
Enter the number of the column to aggregate (0 for *): 1

Available aggregation functions: COUNT, SUM, MIN, MAX, AVG
Enter the aggregation function: COUNT
```

Résultat :

```
Table: drug_dispensing_record_aggregation
╔═══════════════════════════════════════╦═══════════════════════════════════════════════
═════╗
║ drug dispensing record          ║ COUNT(drug dispensing record item)        ║
╠═══════════════════════════════════════╬═══════════════════════════════════════════════
═════╣
║ record_1                        ║ 3                                          ║
║ record_2                        ║ 2                                          ║
║ record_3                        ║ 4                                          ║
╚═══════════════════════════════════════╩═══════════════════════════════════════════════
═════╝
```

3.1.6. MS_SCF : Sélection de classe avec filtre

Ce modèle permet de sélectionner des instances d'une classe ontologique en appliquant un filtre.

Paramètres requis : * `class` : Nom de la classe ontologique à sélectionner * `filterColumn` : Colonne à filtrer (collectée interactivement) * `filterCondition` : Condition de filtrage (collectée interactivement)

Exemple :

```
> !MS_SCF
Enter value for class: http://purl.obolibrary.org/obo/HDRN_0000003

Available columns for filtering:
1. HDRN_0000003_uid
2. HDRN_0000003_HDRN_0000001_dateTime_value
Enter the number of the column to filter on: 2

Examples of filter conditions:
  > '2023-01-01'    - Greater than 2023-01-01
  = '2023-06-15'    - Equal to '2023-06-15'
  BETWEEN '2023-01-01' AND '2023-12-31' - Between two dates
Enter the filter condition: BETWEEN '2023-01-01' AND '2023-12-31'
```

Résultat :

```
Table: HDRN_0000003
╔═══════════════════════════════════════╦═══════════════════╗
║ HDRN human birth temporal information ║ date of birth     ║
╠═══════════════════════════════════════╬═══════════════════╣
║ birth_record_1                        ║ 2023-03-15        ║
╟───────────────────────────────────────╫───────────────────╢
║ birth_record_2                        ║ 2023-07-22        ║
╟───────────────────────────────────────╫───────────────────╢
║ birth_record_3                        ║ 2023-11-05        ║
╚═══════════════════════════════════════╩═══════════════════╝
```

3.1.7. MS_SCPOC : Sélection de classe par propriété d'objet associée à une autre classe

Ce modèle permet de sélectionner des instances d'une classe ontologique liées à des instances d'une autre classe par une propriété d'objet.

Paramètres requis : * `domain` : Nom de la classe ontologique source * `objectProperty` : Nom de la propriété d'objet à utiliser * `range` : Nom de la classe ontologique cible

Exemple :

```
> !MS_SCPOC
Enter value for domain: http://purl.obolibrary.org/obo/PDRO_0000042
Enter value for objectProperty: http://purl.obolibrary.org/obo/BFO_00000051
Enter value for range: http://purl.obolibrary.org/obo/PDRO_0000041
```

Résultat :

```
Table: PDRO_0000042_BFO_00000051_PDRO_0000041
╔═══════════════════════════╦═══════════════════════════╗
║ drug dispensing record    ║ drug dispensing record item ║
╠═══════════════════════════╬═══════════════════════════╣
║ record_1                  ║ item_1                    ║
║ record_1                  ║ item_2                    ║
║ record_1                  ║ item_3                    ║
║ record_2                  ║ item_4                    ║
║ record_2                  ║ item_5                    ║
╚═══════════════════════════╩═══════════════════════════╝
```

3.2. Modèles itératifs (MI_*)

Les modèles itératifs permettent d'explorer les chemins entre concepts ontologiques.

3.2.1. MI_AP : Tous les chemins possibles entre deux classes

Ce modèle permet de trouver tous les chemins possibles entre deux classes ontologiques.

Paramètres requis : * `from` : Nom de la classe ontologique source * `to` : Nom de la classe ontologique cible

Exemple :

```
> !MI_AP
Enter value for from: http://purl.obolibrary.org/obo/HOSO_0000060
Enter value for to: http://purl.obolibrary.org/obo/HOSO_0000062

Available paths:
1. HOSO_0000060 (public provincial health insurance record) -> IAO_0000136 (is about) ->
HOSO_0000062 (primary human health insurance beneficiary)
2. HOSO_0000060 (public provincial health insurance record) -> IAO_0000136 (is about) ->
HOSO_0000019 (public provincial health identifier) -> IAO_0000136 (is about) ->
HOSO_0000062 (primary human health insurance beneficiary)
Select a path (1-2): 1
```

Résultat :

```
Table: HOSO_0000060_IAO_0000136_HOSO_0000062
╔═══════════════════════════════════════════════════════╦═══════════════════════════════════════════════════════
═══════════════════════════╗
║ public provincial health insurance record   ║ primary human health insurance
beneficiary     ║
╠═══════════════════════════════════════════════════════╬═══════════════════════════════════════════════════════
═══════════════════════════╣
║ Record 001                                  ║ Beneficiary 001
║
║ Record 002                                  ║ Beneficiary 002
║
║ Record 003                                  ║ Beneficiary 003
║
╚═══════════════════════════════════════════════════════╩═══════════════════════════════════════════════════════
═══════════════════════════╝
```

3.2.2. MI_SP : Chemin le plus court entre deux classes

Ce modèle permet de trouver le chemin le plus court entre deux classes ontologiques.

Paramètres requis : * `from` : Nom de la classe ontologique source * `to` : Nom de la classe ontologique cible

Exemple :

```
> !MI_SP
Enter value for from: http://purl.obolibrary.org/obo/HOSO_0000060
Enter value for to: http://purl.obolibrary.org/obo/HOSO_0000062
```

Résultat :

```
Shortest path: HOSO_0000060 (public provincial health insurance record) -> IAO_0000136
(is about) -> HOSO_0000062 (primary human health insurance beneficiary)

Table: HOSO_0000060_IAO_0000136_HOSO_0000062
╔═══════════════════════════════════════════════════════╦═══════════════════════════════════════════════════════
═══════════════════════════╗
```

```
╠ public provincial health insurance record   ╠ primary human health insurance
beneficiary     ╠
╠═══════════════════════════════════════════════╬═══════════════════════════════
═══════════════╣
╠ Record 001                                    ╠ Beneficiary 001
╠
╠ Record 002                                    ╠ Beneficiary 002
╠
╠ Record 003                                    ╠ Beneficiary 003
╠
╚═══════════════════════════════════════════════╩═══════════════════════════════
═══════════════╝
```

3.2.3. MI_PPO : Chemin par propriété d'objet spécifique

Ce modèle permet de trouver un chemin entre deux classes ontologiques en utilisant une propriété d'objet spécifique.

Paramètres requis : * `from` : Nom de la classe ontologique source * `to` : Nom de la classe ontologique cible * `objectProperty` : Nom de la propriété d'objet à utiliser

Exemple :

```
> !MI_PPO
Enter value for from: http://purl.obolibrary.org/obo/HOSO_0000060
Enter value for to: http://purl.obolibrary.org/obo/HOSO_0000062
Enter value for objectProperty: http://purl.obolibrary.org/obo/IAO_0000136
```

Résultat :

```
Path: HOSO_0000060 (public provincial health insurance record) -> IAO_0000136 (is about)
-> HOSO_0000062 (primary human health insurance beneficiary)

Table: HOSO_0000060_IAO_0000136_HOSO_0000062
╔═══════════════════════════════════════════════╦═══════════════════════════════
═══════════════╗
╠ public provincial health insurance record   ╠ primary human health insurance
beneficiary     ╠
╠═══════════════════════════════════════════════╬═══════════════════════════════
═══════════════╣
╠ Record 001                                    ╠ Beneficiary 001
╠
╠ Record 002                                    ╠ Beneficiary 002
╠
╠ Record 003                                    ╠ Beneficiary 003
╠
╚═══════════════════════════════════════════════╩═══════════════════════════════
═══════════════╝
```

3.3. Modèle combinatoire (MC_*)

Le modèle combinatoire permet de combiner plusieurs modèles de requêtes.

3.3.1. MC_COMBINATORIAL : Combinaison de plusieurs modèles

Ce modèle permet de combiner les résultats de plusieurs modèles de requêtes en utilisant des expressions de table communes (CTE) avec la clause WITH en SQL. Cette approche permet d'organiser efficacement des requêtes complexes en les décomposant en parties plus petites et plus gérables.

Paramètres requis : * `subModels` : Liste des sous-modèles à combiner, séparés par des virgules

Exemple :

```
> !MC_COMBINATORIAL
Enter sub-models (comma-separated): MS_SC,MI_SP

Parameters for sub-model: MS_SC
Enter value for MS_SC - class: http://purl.obolibrary.org/obo/HOSO_0000060

Parameters for sub-model: MI_SP
Enter value for MI_SP - from: http://purl.obolibrary.org/obo/HOSO_0000060
Enter value for MI_SP - to: http://purl.obolibrary.org/obo/HOSO_0000062
```

Résultat :

```
Combined results using Common Table Expressions (WITH):

WITH
  records AS (
    SELECT HOSO_0000060_uid as record_id, 'public provincial health insurance record' as
record_label
    FROM HOSO_0000060
  ),
  beneficiaries AS (
    SELECT p.HOSO_0000060_uid as record_id,
           b.HOSO_0000062_uid as beneficiary_id,
           'primary human health insurance beneficiary' as beneficiary_label
    FROM "HOSO_0000060" p
    JOIN "HOSO_0000060_IAO_0000136_HOSO_0000062" j ON j.HOSO_0000060_uid =
p.HOSO_0000060_uid
    JOIN "HOSO_0000062" b ON b.HOSO_0000062_uid = j.HOSO_0000062_uid
  )
SELECT r.record_id, r.record_label, b.beneficiary_id, b.beneficiary_label
FROM records r
LEFT JOIN beneficiaries b ON r.record_id = b.record_id;

Table: Combined_Results
╔═══════════╦═══════════════════════════════════════════╦═══════════════╦═══════════════
═══╦═══════════════════════════════════════╗
║ record_id ║ record_label                              ║ beneficiary_id ║
beneficiary_label                          ║
╠═══════════╬═══════════════════════════════════════════╬═══════════════╬═══════════════
═══╬═══════════════════════════════════════╣
║ 1         ║ public provincial health insurance record ║ 1             ║ primary
human health insurance beneficiary     ║
║ 2         ║ public provincial health insurance record ║ 2             ║ primary
human health insurance beneficiary     ║
║ 3         ║ public provincial health insurance record ║ 3             ║ primary
human health insurance beneficiary     ║
╚═══════════╩═══════════════════════════════════════════╩═══════════════╩═══════════════
═══╩═══════════════════════════════════════╝
```

4. Scénarios d'utilisation réels

4.1. Scénario 1 : Analyse de données médicales

Dans ce scénario, nous utilisons les MCR pour analyser des données médicales en utilisant l'ontologie HDRN (Health Data Research Network).

Objectif : Trouver tous les dossiers d'assurance santé provinciaux et leurs bénéficiaires associés.

Étapes :

1. Utiliser MS_SC pour trouver tous les dossiers d'assurance santé provinciaux :

```
> !MS_SC
Enter value for class: http://purl.obolibrary.org/obo/HOSO_0000060
```

1. Utiliser MI_SP pour trouver le chemin entre les dossiers d'assurance et les bénéficiaires :

```
> !MI_SP
Enter value for from: http://purl.obolibrary.org/obo/HOSO_0000060
Enter value for to: http://purl.obolibrary.org/obo/HOSO_0000062
```

1. Utiliser MS_SCPOC pour trouver les dossiers d'assurance liés à des bénéficiaires spécifiques :

```
> !MS_SCPOC
Enter value for domain: http://purl.obolibrary.org/obo/HOSO_0000060
Enter value for objectProperty: http://purl.obolibrary.org/obo/IAO_0000136
Enter value for range: http://purl.obolibrary.org/obo/HOSO_0000062
```

1. Utiliser MS_SCF pour filtrer les dossiers d'assurance par date :

```
> !MS_SCF
Enter value for class: http://purl.obolibrary.org/obo/HOSO_0000060

Available columns for filtering:
1. HOSO_0000060_uid
2. creation_date
Enter the number of the column to filter on: 2

Enter the filter condition: > '2023-01-01'
```

1. Utiliser MC_COMBINATORIAL pour combiner les résultats :

```
> !MC_COMBINATORIAL
Enter sub-models (comma-separated): MS_SCPOC,MS_SCF

Parameters for sub-model: MS_SCPOC
Enter value for MS_SCPOC - domain: http://purl.obolibrary.org/obo/HOSO_0000060
Enter value for MS_SCPOC - objectProperty: http://purl.obolibrary.org/obo/IAO_0000136
Enter value for MS_SCPOC - range: http://purl.obolibrary.org/obo/HOSO_0000062

Parameters for sub-model: MS_SCF
Enter value for MS_SCF - class: http://purl.obolibrary.org/obo/HOSO_0000060
Enter value for MS_SCF - filterColumn: creation_date
Enter value for MS_SCF - filterCondition: > '2023-01-01'
```

4.2. Scénario 2 : Analyse de données pharmaceutiques

Dans ce scénario, nous utilisons les MCR pour analyser des données pharmaceutiques en utilisant l'ontologie PDRO (Prescription Drug Ontology).

Objectif : Trouver tous les enregistrements de distribution de médicaments et calculer le nombre d'articles par enregistrement.

Étapes :

1. Utiliser MS_SC pour trouver tous les enregistrements de distribution de médicaments :

```
> !MS_SC
Enter value for class: http://purl.obolibrary.org/obo/PDRO_0000042
```

1. Utiliser MS_SCPOC pour trouver les articles associés à chaque enregistrement :

```
> !MS_SCPOC
Enter value for domain: http://purl.obolibrary.org/obo/PDRO_0000042
Enter value for objectProperty: http://purl.obolibrary.org/obo/BFO_00000051
Enter value for range: http://purl.obolibrary.org/obo/PDRO_0000041
```

1. Utiliser MS_SCA pour calculer le nombre d'articles par enregistrement :

```
> !MS_SCA
Enter value for class: http://purl.obolibrary.org/obo/PDRO_0000042

Available columns for aggregation:
1. PDRO_0000042_uid
2. PDRO_0000041_uid
0. * (All columns)
Enter the number of the column to aggregate (0 for *): 2

Available aggregation functions: COUNT, SUM, MIN, MAX, AVG
Enter the aggregation function: COUNT
```

Résultat :

```
Table: drug_dispensing_record_aggregation
╔═══════════════════════════════════╦═══════════════════════════════════════════════════════════════════════════════╗
═══════╗
║ drug dispensing record            ║ COUNT(drug dispensing record item)                                            ║
╠═══════════════════════════════════╬═══════════════════════════════════════════════════════════════════════════════╣
═══════╣
║ record_1                          ║ 3                                                                             ║
║ record_2                          ║ 2                                                                             ║
║ record_3                          ║ 4                                                                             ║
╚═══════════════════════════════════╩═══════════════════════════════════════════════════════════════════════════════╝
═══════╝
```

5. Bonnes pratiques

5.1. Choix du modèle approprié

Pour choisir le modèle le plus approprié :

- Utilisez les modèles simples (MS_*) pour des requêtes directes sur une classe ontologique
- Utilisez les modèles itératifs (MI_*) pour explorer les chemins entre classes ontologiques
- Utilisez les modèles simples avec agrégation (MS_SCA) pour calculer des statistiques
- Utilisez les modèles simples avec filtres (MS_SCF) pour filtrer les résultats
- Utilisez le modèle combinatoire (MC_*) pour des requêtes complexes combinant plusieurs modèles

5.2. Optimisation des requêtes

Pour optimiser les performances des requêtes :

- Limitez le nombre de jointures en choisissant des chemins courts dans les modèles itératifs
- Utilisez des filtres (MS_SCF) pour réduire la quantité de données traitées avant d'appliquer des jointures
- Préférez les agrégations (MS_SCA) sur des colonnes indexées
- Évitez les produits cartésiens dans les requêtes combinatoires (MC_*)
- Utilisez des IRI complets pour identifier précisément les classes et propriétés
- Spécifiez l'UUID d'OntoRel pour éviter les ambiguïtés entre différentes instances d'OntoRel

5.3. Gestion des erreurs

En cas d'erreur :

- Vérifiez que les IRI des classes et des propriétés sont corrects et complets
- Assurez-vous que les chemins entre classes existent dans l'ontologie
- Vérifiez que les conditions de filtrage sont valides pour le type de données de la colonne
- Assurez-vous que l'UUID d'OntoRel est correct
- Consultez les fichiers de journalisation pour plus de détails sur les erreurs
- Vérifiez que les tables et colonnes référencées existent dans la base de données OntoRel

6. Conclusion

Les MCR offrent une approche puissante et flexible pour interroger des données ontologiques. Ils permettent aux utilisateurs de formuler des requêtes en termes ontologiques, sans avoir à connaître les détails du schéma de base de données sous-jacent.

En utilisant les MCR, vous pouvez explorer des données, analyser des relations, construire des requêtes complexes et calculer des statistiques sur vos données ontologiques. Les différents types de modèles (simples, itératifs, combinatoires) vous permettent de répondre à une grande variété de besoins d'interrogation de données.

7. Annexes

7.1. Glossaire

- **MCR** : Modèles de Conception de Requêtes, patrons prédéfinis pour la construction de requêtes ontologiques
- **MS_SC** : Modèle simple pour la sélection d'une classe ontologique
- **MS_SCPO** : Modèle simple pour la sélection d'une classe par propriété d'objet
- **MS_SCPD** : Modèle simple pour la sélection d'une classe par propriété de données
- **MS_SCH** : Modèle simple pour la sélection d'une classe par hiérarchie

- **MS_SCA** : Modèle simple pour la sélection d'une classe avec agrégation
- **MS_SCF** : Modèle simple pour la sélection d'une classe avec filtre
- **MS_SCPOC** : Modèle simple pour la sélection d'une classe par propriété d'objet associée à une autre classe
- **MI_AP** : Modèle itératif pour tous les chemins possibles entre deux classes
- **MI_SP** : Modèle itératif pour le chemin le plus court entre deux classes
- **MI_PPO** : Modèle itératif pour un chemin par propriété d'objet spécifique
- **MC_COMBINATORIAL** : Modèle combinatoire pour la combinaison de plusieurs modèles
- **OntoGraph** : Graphe représentant les relations ontologiques
- **OntoRelGraph** : Graphe représentant les mappings entre ontologie et modèle relationnel
- **OntoRelCat** : Catalogue des mappings entre ontologie et modèle relationnel
- **Onto_class** : Table du catalogue OntoRelCat contenant les informations sur les classes ontologiques
- **Onto_class_axiom** : Table du catalogue OntoRelCat contenant les informations sur les axiomes de classe
- **Onto_data_axiom** : Table du catalogue OntoRelCat contenant les informations sur les axiomes de données
- **Onto_class_inheritance** : Table du catalogue OntoRelCat contenant les informations sur les hiérarchies de classes

8. Guide de dépannage

Cette section présente les problèmes courants rencontrés lors de l'utilisation des MCR et leurs solutions.

8.1. Erreurs courantes et solutions

8.1.1. Erreur : "Class not found in ontology"

Problème : L'IRI de classe spécifié n'existe pas dans l'ontologie.

Solution : 1. Vérifiez l'orthographe de l'IRI 2. Utilisez la commande `list-classes` pour voir les classes disponibles 3. Assurez-vous que l'ontologie est correctement chargée

Exemple :

```
> !MS_SC
Enter value for class: http://purl.obolibrary.org/obo/HOSO_0000060X
Error: Class not found in ontology

> list-classes
Available classes:
- http://purl.obolibrary.org/obo/HOSO_0000060 (public provincial health insurance
record)
- http://purl.obolibrary.org/obo/HOSO_0000062 (primary human health insurance
beneficiary)
...

> !MS_SC
Enter value for class: http://purl.obolibrary.org/obo/HOSO_0000060
```

8.1.2. Erreur : "No path found between classes"

Problème : Aucun chemin n'a été trouvé entre les classes spécifiées dans les modèles itératifs.

Solution : 1. Vérifiez que les classes source et cible existent 2. Utilisez MI_AP pour explorer tous les chemins possibles 3. Considérez l'utilisation de classes intermédiaires

Exemple :

```
> !MI_SP
Enter value for from: http://purl.obolibrary.org/obo/HOSO_0000060
Enter value for to: http://purl.obolibrary.org/obo/PDRO_0000042
Error: No path found between classes

> !MI_AP
Enter value for from: http://purl.obolibrary.org/obo/HOSO_0000060
Enter value for to: http://purl.obolibrary.org/obo/HOSO_0000062

Available paths:
1. HOSO_0000060 (public provincial health insurance record) -> IAO_0000136 (is about) ->
HOSO_0000062 (primary human health insurance beneficiary)
Select a path (1): 1
```

8.1.3. Erreur : "Invalid SQL generated"

Problème : La requête SQL générée est invalide, souvent en raison de problèmes de jointure.

Solution : 1. Vérifiez les mappings entre classes ontologiques et tables relationnelles 2. Utilisez des modèles plus simples (MS_* au lieu de MC_*) 3. Examinez les logs pour identifier la partie problématique de la requête

Exemple :

```
> !MC_COMBINATORIAL
Enter sub-models (comma-separated): MS_SC,MS_SCPO,MS_SCPD
Error: Invalid SQL generated - Column 'HOSO_0000060_uid' not found in table
'PDRO_0000042'

> !MC_COMBINATORIAL
Enter sub-models (comma-separated): MS_SC,MS_SCPO
```

8.2. Vérification de l'environnement

Si vous rencontrez des problèmes persistants, vérifiez les éléments suivants :

1. **Configuration de la base de données** :
   - Vérifiez que la connexion à la base de données est active
   - Assurez-vous que les tables OntoRelCat existent et sont accessibles
   - Vérifiez les permissions d'accès aux tables
2. **Configuration de l'ontologie** :
   - Assurez-vous que l'ontologie est correctement chargée
   - Vérifiez que les mappings entre ontologie et schéma relationnel sont à jour
   - Vérifiez l'UUID d'OntoRel utilisé dans les requêtes
3. **Logs et diagnostics** :
   - Examinez les fichiers de log pour des erreurs détaillées
   - Utilisez l'option `--verbose` pour obtenir plus d'informations de débogage
   - Vérifiez les requêtes SQL générées dans les fichiers exportés

9. Références

- Documentation technique des MCR (SES)
- Documentation française du projet OntoRelQuery (OntoRelQuery_Documentation_FR.adoc)
- Documentation anglaise du projet OntoRelQuery (OntoRelQuery_Documentation_EN.adoc)
- Spécifications des modèles de requêtes (OntoRelQuery_conception.pdf)
- Documentation de l'ontologie HDRN (Health Data Research Network)
- Documentation de l'ontologie PDRO (Prescription Drug Ontology)
- Documentation sur les Common Table Expressions (CTE) : https://www.postgresql.org/docs/current/queries-with.html
- Documentation JOOQ : https://www.jooq.org/doc/