

Évaluation des modèles de conception de requêtes

Rapport d'évaluation des MCR

Table des matières

1	Introduction	2
2	Méthodologie d'évaluation	2
2.1	Cas d'utilisation	2
2.2	Jeu de données	2
2.3	Définition des variables	2
2.4	Interface de commande	3
3	Évaluation des modèles simples	4
3.1	Modèle MS_SCPO : Extraction simple avec propriété objet	5
3.2	Modèle MS_SCPD : Extraction de propriété de données	6
3.3	Modèle MS_SCPOF : Extraction avec filtres	7
3.4	Modèle MS_SCA : Agrégation	8
4	Évaluation des modèles itératifs	8
4.1	Modèle MI_AP : Exploration exhaustive (critère ALL)	9
4.2	Modèle MI_SP : Recherche du plus court chemin (critère MIN)	10
4.3	Modèle MI_PPOF : Exploration avec contraintes	10
5	Évaluation des modèles combinatoires	11
5.1	Modèle MC_UNION : Fusion de parcours	11
6	Conclusion	12

1 Introduction

Ce rapport présente une évaluation des [Modèles de Conception de Requêtes \(MCRs\)](#) en contexte clinique réalisée dans le cadre du projet OntoRelQuery. L'évaluation porte sur huit modèles représentatifs des trois catégories taxonomiques définies dans le chapitre 3 du mémoire : modèles simples, modèles itératifs et modèles combinatoires.

La section 2 décrit le cas d'utilisation et l'interface de commande utilisée pour la formulation des demandes. Les sections 3, 4 et 5 détaillent respectivement l'évaluation des trois catégories de modèles.

2 Méthodologie d'évaluation

2.1 Cas d'utilisation

L'évaluation s'inscrit dans un projet d'aide à la décision visant à améliorer les soins palliatifs et de fin de vie pour les patients à risque élevé de mortalité à un an. Ce projet analyse l'utilité de modèles prédictifs exploitant des données recueillies régulièrement : informations démographiques, informations d'hospitalisation et préférences thérapeutiques.

2.2 Jeu de données

L'évaluation s'appuie sur l'ontologie [Ontologie de prédiction de la mortalité des patients hospitalisés \(MPHPO\)](#)¹ développée pour définir la sémantique des données collectées. Cette ontologie structure 13 variables cliniques utilisées pour entraîner 4 modèles d'apprentissage automatique. Un [Modèle ontologique-relationnel \(OntoRel\)](#) a été alimenté à partir de [Medical Information Mart for Intensive Care \(MIMIC-IV\)](#)², contenant des données cliniques réelles.

2.3 Définition des variables

Chaque [MCR](#) évalué extrait des fragments de ces 13 variables pour démontrer sa capacité fonctionnelle. Le tableau 1 présente la correspondance entre les modèles évalués et les variables cliniques du projet. Cette évaluation porte sur huit modèles couvrant les opérations fondamentales des [MCRs](#), telles que définies dans le chapitre 3 du mémoire.

1. MPHPO - Mortality Prediction for Hospitalized Patients Ontology. Voir : <https://github.com/OpenLHS/MPHPO>

2. MIMIC-IV - Medical Information Mart for Intensive Care. Base de données anonymisée contenant 431 231 admissions hospitalières entre 2008 et 2019. Voir : <https://mimic.mit.edu/>

Il est important de noter que les **MCRs** évalués n’implémentent pas les variables cliniques au complet. Chaque modèle extrait plutôt des fragments spécifiques correspondant à des sous-parcours ou à des composantes particulières d’une variable complète. Par exemple, la variable 1 (mortalité à un an) nécessite 3 parcours ontologiques distincts pour être calculée entièrement, mais le modèle MI_AP n’en implémente qu’un seul (Parcours 2 : Patient vers Hospitalisation). Cette approche permet de mettre en évidence et d’évaluer les capacités fonctionnelles de chaque **MCR** à partir de cas d’usage ciblés.

TABLE 1 – Correspondance entre modèles évalués et variables cliniques

Variable	Définition	MCR évalué
Variable 1	Mortalité à un an	MI_AP
Variable 2	Âge du patient (Date de naissance)	MS_SCPD
Variable 4	Nombre de visites aux urgences (filtrées)	MS_SCPOF, MS_SCA
Variable 7	Type d’admission (niveau d’urgence)	MS_SCPO
Variable 8	Service d’admission (spécialité médicale)	MI_SP, MI_PPOF
Variables 1+2	Fusion : Mortalité à un an et Âge	MC_UNION

2.4 Interface de commande

L’évaluation s’effectue via l’interface en ligne de commande du prototype **Système d’interrogation ontologique relationnelle (OntoRelQuery)**. La figure 1 présente l’écran d’accueil après initialisation. Le système affiche le répertoire de sortie, les paramètres de connexion **Système de gestion de bases de données (SGBD)**, l’identifiant de l’**OntoRel** avec ses schémas (MPHPO et ontorelcat_api_ontorelquery pour le catalogue), et les chemins des graphes **Grappe orienté représentant les relations ontologiques (OntoGraph)** et **Grappe orienté représentant les relations ontologiques relationnelles (OntoRelGraph)**.

```

# OntoRelQuery #

Welcome to OntoRelQuery
A tool for querying relational data using ontology-based models
Version: 1.0

Output directory
• Path: /Users/gaim2203/IdeaProjects/OntoRelQuery/

Database connection
• Database: mimic_iv
• User: gaim2203
• Host: localhost
• Port: 5432

OntoRel information
• OntoRel UUID: 8f937ca7-8bdc-40da-9548-ece5eca439cb
• OntoRel schema: MPHP0
• OntoRelCat schema: ontorelcat_api_ontorelquery

Graph files
• OntoGraph: test-data/_ontologies/MPHP0/config00/20240820-1631/Graphs/OntoGraph.dot
• OntoRelGraph: test-data/_ontologies/MPHP0/config00/20240820-1631/Graphs/OntoRelGraph.dot

Available commands:
mcr-list - List all available query models
execute --type MODEL_NAME - Execute a specific query model
!MODEL_NAME - Shortcut to execute a model (e.g., !MS_SC)
clear - Clear the console
help - Display help information
exit - Exit the application

Type a command to begin:
You can now use the following subcommands:
Usage: [COMMAND]
Commands:
  help      Display help information about the specified command.
  mcr-list  List all available query models with descriptions and required
            options.
  execute   Execute a query model.
  clear     Clear the console.
Or use '!' followed by a model name or index to execute it directly (e.g., !MS_SC or !1)
> █

```

FIGURE 1 – Interface principale de l’outil OntoRelQuery

Les commandes disponibles sont :

- `mcr-list` : lister les modèles implémentés,
- `execute -type MODEL_NAME` : exécuter un modèle,
- `clear` : réinitialiser les paramètres de recherche,
- `exit` : quitter l’application.

3 Évaluation des modèles simples

Cette section évalue quatre modèles simples qui illustrent les opérations de base : extraction simple de classes connectées par propriétés objet (MS_SCPO), extraction de

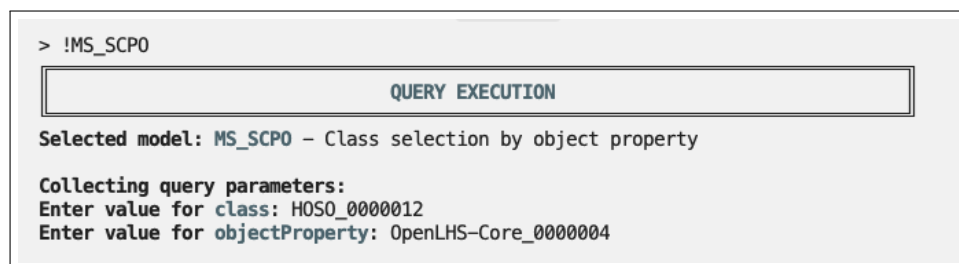
propriétés de données (MS_SCPD), application de filtres (MS_SCPOF), et agrégation (MS_SCA).

3.1 Modèle MS_SCPO : Extraction simple avec propriété objet

Demande. Identifier les entités qui spécifient le niveau d'urgence de l'admission hospitalière. L'objectif est d'extraire les identifiants uniques des spécifications de visite qui dirigent les visites cliniques, incluant les spécifications de niveau d'urgence qui héritent de cette classe.

Correspondance ontologique. Ce besoin implique la classe HOSO_0000012 (healthcare organization clinical visit) liée par la propriété objet `OpenLHS-Core_0000004` (is directed by) vers deux classes cibles : HOSO_0000003 (healthcare service organism specification) et HOSO_0000074 (clinical visit agreement specification). La classe MPHPO_0000007 hérite de HOSO_0000074, permettant d'extraire spécifiquement les niveaux d'urgence via ce parcours.

Paramètres. Deux paramètres requis : classe source et propriété objet. La figure 2 présente la collecte interactive des paramètres.



The screenshot shows a terminal window with the following content:

```
> !MS_SCPO
```

QUERY EXECUTION

Selected model: **MS_SCPO** – Class selection by object property

Collecting query parameters:
Enter value for **class**: HOSO_0000012
Enter value for **objectProperty**: OpenLHS-Core_0000004

FIGURE 2 – MS_SCPO : Collecte des paramètres

Construction. La vérification du catalogue identifie trois relations : HOSO_0000012 et deux axiomes vers HOSO_0000003 et HOSO_0000074 (figure 3).



FIGURE 3 – MS_SCPO : Vérification des relations dans le catalogue

La requête [Langage de requêtes structurées \(SQL\)](#) est formée de deux jointures avec une projection sur les trois identifiants uniques (figure 4).

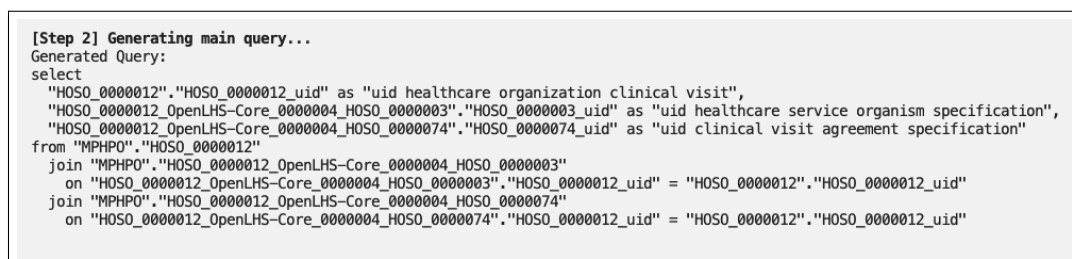


FIGURE 4 – MS_SCPO : Requête SQL générée

3.2 Modèle MS_SCPD : Extraction de propriété de données

Demande. Récupérer les dates de naissance des patients hospitalisés. L'objectif est d'extraire les identifiants uniques associés aux informations de naissance et de les lier à leurs valeurs de date correspondantes, permettant ainsi le calcul de l'âge du patient au moment de l'admission.

Correspondance ontologique. Ce besoin implique la classe MPHPO_0000009 (MPHPO human birth date) liée par la propriété de données Ontorel-Core_0000004 (has temporal value) vers le type de données Ontorel-Core_0000001 (ISO8601DateTime).

Résultats. La requête projette l'[Identifiant unique universel \(UUID\)](#) patient et la valeur temporelle (figure 5).

```
[Step 2] Generating main query...
Generated Query:
select
  "MPHPO_0000009"."MPHPO_0000009_uid" as "uid MPHPO human birth date",
  "MPHPO_0000009_Ontorel-Core_0000004_Ontorel-Core_0000001"."MPHPO_0000009_Ontorel-Core_0000004_Ontorel-Core_0000001_Ontorel" as "h
as temporal value"
from "MPHPO"."MPHPO_0000009"
join "MPHPO"."MPHPO_0000009_Ontorel-Core_0000004_Ontorel-Core_0000001"
  on "MPHPO_0000009_Ontorel-Core_0000004_Ontorel-Core_0000001"."MPHPO_0000009_uid" = "MPHPO_0000009"."MPHPO_0000009_uid"
```

FIGURE 5 – MS_SCPD : Requête SQL générée

3.3 Modèle MS_SCPOF : Extraction avec filtres

Demande. Identifier les visites cliniques associées à des organisations spécifiques et possédant des accords de visite valides. L’objectif est de filtrer les visites cliniques pour ne retenir que celles liées à certaines organisations prédéfinies.

Filtrage. Deux conditions sont appliquées : vérification de la présence d’un accord de visite (IS NOT NULL) et sélection d’organisations spécifiques via leurs identifiants (figures 6 et 7).

```
=== Multiple Filter Conditions ===

=== Filter Condition Builder ===
Build your WHERE clause with multiple conditions.
Press Enter at column selection to finish.

--- Condition #1 ---

Available columns:
  1. HOSO_0000074_uid
  2. HOSO_0000003_uid
  3. HOSO_0000012_uid
Select column number (Enter to finish): 1

Supported operators:
  1. = (equal)
  2. != (not equal)
  3. > (greater than)
  4. >= (greater or equal)
  5. < (less than)
  6. <= (less or equal)
  7. LIKE (pattern match)
  8. IN (in list)
  9. IS NULL (is null)
  10. IS NOT NULL (is not null)
Select operator number: 10
✓ Added: "MPHPO"."HOSO_0000012_OpenLHS-Core_0000004_HOSO_0000074"."HOSO_0000074_uid" IS NOT NULL

Add another condition? (y/n): y
```

FIGURE 6 – MS_SCPOF : Construction de la première condition

```

--- Condition #2 ---

Available columns:
1. HOSO_0000074_uid
2. HOSO_0000003_uid
3. HOSO_0000012_uid
Select column number (Enter to finish): 2

Supported operators:
1. = (equal)
2. != (not equal)
3. > (greater than)
4. >= (greater or equal)
5. < (less than)
6. <= (less or equal)
7. LIKE (pattern match)
8. IN (in list)
9. IS NULL (is null)
10. IS NOT NULL (is not null)
Select operator number: 8
Tip: Example: ('a','b','c') or (1,2,3)
Enter value: ('5a1757f0-db12-456f-8643-7e8a94ab89e1','f4af628d-b38f-4afd-86cb-4f39c0da4df8')

Logical connector:
1. AND (all conditions must match)
2. OR (at least one condition must match)
Select (default: AND): 1
✓ Added: "MPHP0"."HOSO_0000012_OpenLHS-Core_0000004_HOSO_0000003"."HOSO_0000003_uid" IN ('5a1757f0-db12-456f-8643-7e8a94ab89e1','f4af628d-b38f-4afd-86cb-4f39c0da4df8')

Current WHERE clause:
"MPHP0"."HOSO_0000012_OpenLHS-Core_0000004_HOSO_0000074"."HOSO_0000074_uid" IS NOT NULL AND "MPHP0"."HOSO_0000012_OpenLHS-Core_0000004_HOSO_0000003"."HOSO_0000003_uid" IN ('5a1757f0-db12-456f-8643-7e8a94ab89e1','f4af628d-b38f-4afd-86cb-4f39c0da4df8')

```

FIGURE 7 – MS_SCPOF : Construction de la deuxième condition

3.4 Modèle MS_SCA : Agrégation

Demande. Dénombrer le nombre total d'identifiants patients distincts présents dans la base de données. L'objectif est d'appliquer une fonction d'agrégation pour compter les patients uniques.

Configuration. Après sélection de la classe source, l'utilisateur choisit la colonne et la fonction d'agrégation COUNT (figure 8).

```

Additional Aggregation Options:
Available columns for aggregation:
1. IOIO_0000014_uid
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
Aggregation options collected successfully.

```

FIGURE 8 – MS_SCA : Application de l'agrégation

4 Évaluation des modèles itératifs

Cette section évalue trois modèles itératifs qui explorent des chemins dans l'ontologie : exploration exhaustive (MI_AP), recherche du plus court chemin (MI_SP), et

exploration avec contraintes de propriétés et filtres (MI_PPOF).

4.1 Modèle MI_AP : Exploration exhaustive (critère ALL)

Demande. Explorer tous les chemins possibles qui permettent de relier les hospitalisations aux identifiants des patients concernés. L'objectif est de révéler l'ensemble complet des chemins sémantiques. Ce parcours est essentiel pour calculer la mortalité à un an en permettant de relier chaque hospitalisation au patient correspondant.

Correspondance ontologique. Ce besoin implique un parcours depuis la classe HOSO_0000031 (hospitalization) vers la classe IOIO_0000014 (human identifier). Le chemin traverse cinq classes intermédiaires en utilisant les propriétés isa (héritage), OpenLHS-Core_0000004 (is directed by), IAO_0000235 (denoted by), et IAO_0000219 (denotes).

Sélection. Un seul chemin détecté, reliant chaque hospitalisation à l'identifiant unique du patient concerné (figure 9).



FIGURE 9 – MI_AP : Chemin détecté

Construction. La requête génère 8 jointures reliant 9 tables (figure 10).

```

[Step 2] Generating main query...
Generated Query:
select
  "HOSO_0000031"."HOSO_0000031_uid" as "uid hospitalization",
  "HOSO_0000012"."HOSO_0000012_uid" as "uid healthcare organization clinical visit",
  "HOSO_0000012_OpenLHS-Core_0000004_HOSO_0000003"."HOSO_0000003_uid" as "uid healthcare service organism specification",
  "HOSO_0000137_IAO_0000235_HOSO_0000003"."HOSO_0000137_uid" as "uid human clinical visit specified patient",
  "NCBITaxon_9606"."NCBITaxon_9606_uid" as "uid Homo sapiens",
  "IOIO_0000014_IAO_0000219_NCBITaxon_9606"."IOIO_0000014_uid" as "uid human identifier"
from "MPHP0"."HOSO_0000031"
join "MPHP0"."HOSO_0000012"
  on "HOSO_0000031"."HOSO_0000031_uid" = "HOSO_0000012"."HOSO_0000012_uid"
join "MPHP0"."HOSO_0000012_OpenLHS-Core_0000004_HOSO_0000003"
  on "HOSO_0000012_OpenLHS-Core_0000004_HOSO_0000003"."HOSO_0000012_uid" = "HOSO_0000012"."HOSO_0000012_uid"
join "MPHP0"."HOSO_0000003"
  on "HOSO_0000012_OpenLHS-Core_0000004_HOSO_0000003"."HOSO_0000003_uid" = "HOSO_0000003"."HOSO_0000003_uid"
join "MPHP0"."HOSO_0000137_IAO_0000235_HOSO_0000003"
  on "HOSO_0000137_IAO_0000235_HOSO_0000003"."HOSO_0000003_uid" = "HOSO_0000003"."HOSO_0000003_uid"
join "MPHP0"."HOSO_0000137"
  on "HOSO_0000137_IAO_0000235_HOSO_0000003"."HOSO_0000137_uid" = "HOSO_0000137"."HOSO_0000137_uid"
join "MPHP0"."NCBITaxon_9606"
  on "HOSO_0000137"."HOSO_0000137_uid" = "NCBITaxon_9606"."NCBITaxon_9606_uid"
join "MPHP0"."IOIO_0000014_IAO_0000219_NCBITaxon_9606"
  on "IOIO_0000014_IAO_0000219_NCBITaxon_9606"."NCBITaxon_9606_uid" = "NCBITaxon_9606"."NCBITaxon_9606_uid"
join "MPHP0"."IOIO_0000014"
  on "IOIO_0000014_IAO_0000219_NCBITaxon_9606"."IOIO_0000014_uid" = "IOIO_0000014"."IOIO_0000014_uid"

```

FIGURE 10 – MI_AP : Requête SQL générée

4.2 Modèle MI_SP : Recherche du plus court chemin (critère MIN)

Demande. Trouver le chemin le plus court qui relie les identifiants de services médicaux aux organisations de santé. L'objectif est de minimiser le nombre de jointures nécessaires.

Sélection automatique. Un seul chemin détecté générant une requête avec 2 jointures entre 3 relations (figure 11).

```

PATH SELECTION

Automatically selecting shortest path...
Finding paths between HOSO_0000084 and HOSO_0000008...
Shortest path automatically selected:
healthcare physician service identifier (HOSO_0000084) -> (healthcare physician service identifier denotes healthcare organization (HOSO_0000084_IAO_0000219_HOSO_0000008)) -> healthcare organization (HOSO_0000008)
Path selected successfully.
Building query with 3 tables...

```

FIGURE 11 – MI_SP : Chemin le plus court détecté

4.3 Modèle MI_PPOF : Exploration avec contraintes

Demande. Identifier les services médicaux associés à des organisations de santé spécifiques en imposant une contrainte de navigation stricte : le parcours doit obligatoirement utiliser la relation denotes. Des filtres sont appliqués pour sélectionner des organisations ciblées.

Sélection. Deux chemins disponibles. Choix du premier avec HOSO_0000079 (figure 12).

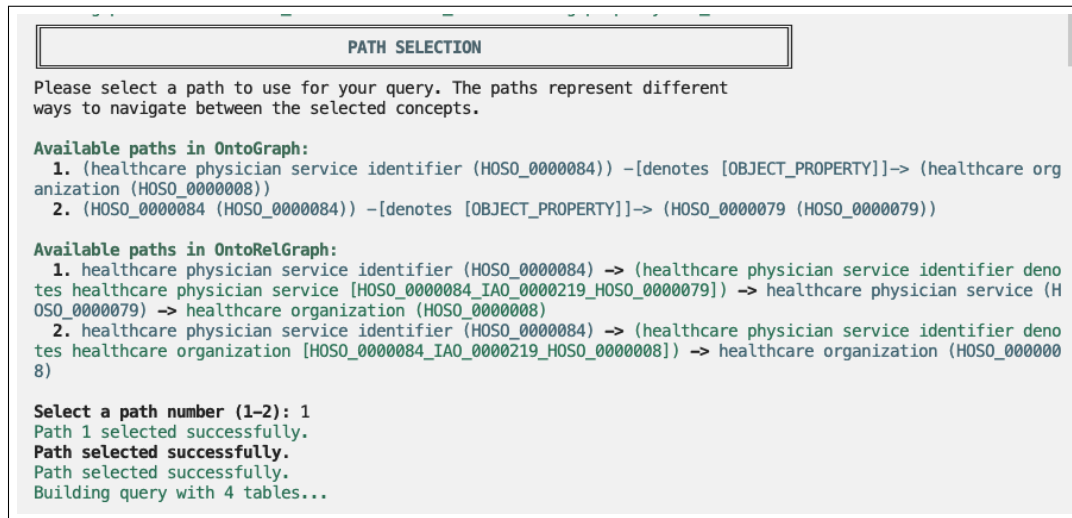


FIGURE 12 – MI_PPOF : Chemins détectés

5 Évaluation des modèles combinatoires

Cette section évalue un modèle combinatoire qui fusionne deux parcours distincts à l'aide de l'opérateur UNION.

5.1 Modèle MC_UNION : Fusion de parcours

Demande. Fusionner deux parcours distincts depuis la classe représentant le patient spécifié d'une visite clinique : l'un menant vers les spécifications organisationnelles, l'autre vers les identifiants personnels du patient. L'objectif est de créer une vue unifiée des patients tout en éliminant les doublons.

Correspondance ontologique. Deux parcours depuis la classe commune HOSO_0000137 (human clinical visit specified patient). Les deux sous-requêtes sont encapsulées et fusionnées par l'opérateur UNION sur la colonne commune, produisant une liste unique de patients sans doublons.

Analyse. Colonne commune détectée: uid human clinical visit specified patient (figure 13).

```
Analyzing common columns...
Found 1 common column(s):
- uid human clinical visit specified patient ("uid human clinical visit specified patient")
```

FIGURE 13 – MC_UNION : Identification de la colonne commune

Construction. Deux sous-requêtes regroupées à l'aide de UNION (figure 14).

QUERY CONSTRUCTION

[Step 1] Combined query generated

Generated Query:

```
select "subquery"."uid human clinical visit specified patient" as "uid human clinical visit specified patient"
from (
  select
    "HOSO_0000137"."HOSO_0000137_uid" as "uid human clinical visit specified patient",
    "HOSO_0000137_IAO_0000235_HOSO_0000003"."HOSO_0000003_uid" as "uid healthcare service organism specification"
  from "MPHP0"."HOSO_0000137"
  join "MPHP0"."HOSO_0000137_IAO_0000235_HOSO_0000003"
    on "HOSO_0000137_IAO_0000235_HOSO_0000003"."HOSO_0000137_uid" = "HOSO_0000137"."HOSO_0000137_uid"
  join "MPHP0"."HOSO_0000003"
    on "HOSO_0000137_IAO_0000235_HOSO_0000003"."HOSO_0000003_uid" = "HOSO_0000003"."HOSO_0000003_uid"
) as "subquery"
union
select "subquery"."uid human clinical visit specified patient" as "uid human clinical visit specified patient"
from (
  select
    "HOSO_0000137"."HOSO_0000137_uid" as "uid human clinical visit specified patient",
    "NCBITaxon_9606"."NCBITaxon_9606_uid" as "uid Homo sapiens",
    "IOIO_0000014_IAO_0000219_NCBITaxon_9606"."IOIO_0000014_uid" as "uid human identifier"
  from "MPHP0"."HOSO_0000137"
  join "MPHP0"."NCBITaxon_9606"
    on "HOSO_0000137"."HOSO_0000137_uid" = "NCBITaxon_9606"."NCBITaxon_9606_uid"
  join "MPHP0"."IOIO_0000014_IAO_0000219_NCBITaxon_9606"
    on "IOIO_0000014_IAO_0000219_NCBITaxon_9606"."NCBITaxon_9606_uid" = "NCBITaxon_9606"."NCBITaxon_9606_uid"
  join "MPHP0"."IOIO_0000014"
    on "IOIO_0000014_IAO_0000219_NCBITaxon_9606"."IOIO_0000014_uid" = "IOIO_0000014"."IOIO_0000014_uid"
) as "subquery"
```

FIGURE 14 – MC_UNION : Requête SQL générée

6 Conclusion

Ce rapport a présenté l'évaluation de huit modèles de conception de requêtes représentatifs des trois catégories taxonomiques. L'évaluation s'est effectuée via l'interface en ligne de commande d'OntoRelQuery sur des fragments de variables cliniques issues d'un projet de prédiction de mortalité.