	LRA Technical Model Infrastructure Specification Part 5: Query Specification			
	Programme	Informatics Data Standards	Document Record ID Key	
	Sub-Prog / Project	Logical Record Architecture for Health and Social Care	NPFIT-FNT-TO-DPM-1025.02	
	Prog. Director	D. Perry	Status	Draft
	Owner	S. Bentley / L. Sato	Version	0.3
	Author	R.Q. Siddiqui	Version Date	26 Mar 2010

Logical Record Architecture for Health and Social Care Technical Model Infrastructure Specification Part 5: Query Specification

Amendment History:

Version	Date	Amendment History
0.1	16 Mar 2010	First draft for internal review
0.2	25 Mar 2010	Updates and addition of renal example
0.3	26 Mar 2010	Final amendments and addition of Recommendations section

Forecast Changes:

Anticipated Change	When

Reviewers:

This document must be reviewed by the following:<author to indicate reviewers>

Name	Signature	Title / Responsibility	Date	Version
Steven Bentley		Acting Head, Logical Record Architecture		
Nicholas Oughtibridge		Lead Architect		

Approvals:

This document must be approved by the following: <author to indicate approvers>

Name	Signature	Title / Responsibility	Date	Version
S. Bentley		Acting Head, Logical Record Architecture		

Distribution:

The document should be distributed to the LRA Team for review.

Document Status:

This is a controlled document.

Whilst this document may be printed, the electronic version maintained in FileCM is the controlled copy. Any printed copies of the document are not controlled.

Related Documents:

These documents will provide additional information.

Ref no	Doc Reference Number	Title	Version
1	NPFIT-SHR-QMS-PRP-0015	Glossary of Terms Consolidated.doc	
2	NPFIT-FNT-TO-DPM-0911.13	LRA For Health and Social Care: Artefacts Overview	2.1
3	NPFIT-FNT-TO-DPM-0950	Evaluation of Technical Model Query Requirements	1.0
4	NPFIT-FNT-TO-DPM-0935.01	LRA Technical Model Infrastructure	0.5

		Specification Part 1: Care Components	
5	NPFIT-FNT-TO-DPM-1001.02	LRA Technical Model Developer's Manual Part 1: Model Design	0.6.1
6	NPFIT-FNT-TO-DPM-0936.01	Logical Record Architecture for Health and Social Care: Data Types Specification	0.3
7	https://svn.connectingforhealth.nhs.uk/svn/public/lra/TRUNK/prod_env/ref/ISO_21090/	Health informatics – Harmonized data types for information interchange	ISO 21090

Glossary of Terms:

List any new terms created in this document. Mail the NPO Quality Manager to have these included in the master glossary above [1].

Term	Acronym	Definition
	BMI	Body Mass Index
	CO	Coded Ordinals
	DH	Department of Health
	DOB	Date of Birth
	EHR	Electronic Health Record
	GUID	Globally Unique Identifier
	IC	Information Centre
	INT	Integer
	LRA	Logical Record Architecture
	MO	Monetary Amount
	PQ	Physical Quantity data type (ISO 21090 data type)
	R3	LRA Release 3
	TS	Time Series data type (ISO 21090 data type)
	UCUM	Unified Code for Units of Measure
	URR	Urea Reduction Ratio

Contents

1	About this Document	6
	Purpose	6
	Audience	6
2	Introduction	6
3	Scope and Assumptions	6
	3.1 Scope	6
	3.2 Assumptions	7
4	LRA Query Model Specification	8
	4.1 Package lra.technical	8
	4.1.1 Class AvailableShareableEHR	8
	4.1.2 Class ShareableEhrSystem	9
	4.1.3 Class SharedDemographicService	9
	4.2 Package lra.technical.query	10
	4.2.1 Abstract Class AbstractQuery	10
	4.2.2 Class AggregatedResultSet	11
	4.2.3 Class DemographicSelectionQuery	11
	4.2.4 Class PopulationQuery	12
	4.2.5 Class QueryParameter	13
	4.2.6 Class SubjectOfCareEhrQuery	13
5	LRA Query Grammar	14
	5.1 Logical and Mathematical Operations	15
	5.1.1 Logical Operators	15
	5.1.2 Mathematical Operators	15
6	Query Format	16
	6.1 Query Definition	16
7	Query Methods	18
	7.1 Computable Query Expression	20
	7.2 Temporal Queries	21
8	Query Examples	22
	8.1 Demographic Selection Queries	25
	8.1.1 Direct Queries	25
	8.2 Subject of Care EHR Queries	28
	8.2.1 Direct Queries	28
	8.2.2 Derived Operation Queries	41
	8.2.3 Direct Inference Queries	44
	8.3 Population Queries	45
	8.3.1 Direct Queries	45
	8.3.2 Derived Operation Queries	48
9	Recommendations	50

A Appendix	51
A.1 Patient Medication Query Illustration.....	51
A.2 Population Obesity Query Illustration	53
A.3 Population Haemodialysis Session Information Query Illustration.....	55
A.4 LRA Technical Query Requirements Summary	57

1 About this Document

Purpose

The purpose of this document is to formally specify the framework for constructing queries against the LRA Interface models (i.e. constrained LRA Technical Models). The document itself forms part of the LRA Infrastructure Specification although the artefacts produced by the document form part of the LRA Interface artefacts [1].

Audience

The intended audience of this specification is any individual, group or organisation involved in the development or use of the LRA.

2 Introduction

The Query Specification document is based on the *Evaluation of LRA Technical Model Query Requirements* document [2]. The purpose of the Query Requirements document was to determine a set of requirements that must be satisfied in order to enable the LRA Technical Models and any formalism used to specify computable query and inference expressions to represent fully the types of query used to retrieve data from care records.

The Technical Model Query Specification is an explicit set of requirements that need to be satisfied by the LRA Technical Model. Therefore, if the Technical Model does not meet one or more of the business requirements then it might require relevant modifications or additions to the model(s).

The Query Specification document aims at providing a 'grammar' i.e. a logical set of rules to be applied to the generation of query expressions independent of its implementation language. In this respect, further query implementation artefacts might be produced based on the Query Specification document post R3. Any Technical Query Requirement specified in [2] that is met by the query specification is indicated in the format [QUERY n] where $n = \{1, 2, \dots, 67\}$.

3 Scope and Assumptions

3.1 Scope

The scope of the LRA Query Specification includes queries which may be expressed to meet the requirements for primary use (i.e. to support direct care of the patient or service user) or secondary use (i.e. for analyses). Each query attempts to satisfy one or more technical requirements as stated in the LRA Query Requirements document [2]. In addition, each query also attempts to satisfy a specific business requirement for which the query is being defined.

The scope of the LRA Query Specification is presented in Table 1 below. LRA queries are defined both on the basis of the subject(s) of care being queried as well as on the basis of the functions applied on the query. Therefore, there are three main kinds of queries viz. Patient, linked (related) patients, and population level queries as shown in Table 1. Each of these three queries types might require one or more functions to be performed to satisfy the query requirement. For instance, a patient level query might require a direct query of his/her EHR followed by some derived operation as well as an inference on the data. E.g. to find out whether a patient has obesity it could either be 'directly' queried from the patient EHR or the information could be 'inferred' by deriving the BMI through an 'operation' on the patient weight and height.

Patient-linked queries are currently out of scope for R3 due to lack of clearly defined clinical examples and their potential complexity. However, the LRA Technical Model has the potential to be modelled in this domain and therefore should be possible to construct patient-linked queries such as queries based on mother and child record links.

In scope: ✓ / ✗ Example	Direct Query	Derived operation	Direct inference	Derived inference
Patient	✓ Height	✓ BMI	✓ Obesity	✗ ICD Coding
Linked (related) Patients	✓ Foetal crown rump length	✓	✓ Gestation stage	✗
Population	✓ All vaccinated patients	✓ Count/average	✓ Herd immunity	✗

Table 1: LRA Query Scope

Another area that has been assumed but not specified for R3 is querying the Demographics Model. At present, the LRA Demographics Model is still at a level of immaturity to enable complex queries to be performed. However, post-R3 there is plan to develop and stabilise the Demographics Model thereby enabling a more detailed demographics query specification to be produced.

3.2 Assumptions

The following assumptions have been made to provide a holistic view of the logical queries:

- (a) Virtual Shareable EHR System - The notion of a virtual system which holds the EHRs of the entire patient population is assumed. This logical separation of a single shareable EHR system helps to abstract away from the physical fragmented representation of various EHR systems. LRA does not assume the storage, organisation or method of accessing patients' or service users' care records by an EHR system.

The virtual EHR system is capable of accessing the available shareable care record content (or parts thereof) of one or more patients or service users for the purpose of specifying one or more LRA Queries. For the purpose of defining LRA Queries, each patient or service user known to the system has at the most one available shareable EHR.

- (b) Virtual Shared Demographics Service – Due to the current immaturity of the LRA Demographics Model, all demographic queries are made to an assumed logical demographics service. For the purposes of query specification, the demographic content to which the system has access to is considered to include any necessary information that conforms to the LRA Participations Reference Model (as defined in the Reference Model package `Ira::technical::participations`).
- (c) Use of EHR system identifiers – All queries are defined using unique system identifiers of patients or service users. These system identifiers are internal to the EHR system and are separate from the business identifiers such as the NHS number which uniquely identify the patient. The LRA Query Specification currently follows the EN13606 approach of separating the system identifiers from the business identifiers and as such does not query on the basis of business identifiers (ids) at any time or return demographic information. The LRA does not

propose how a physical EHR system may reconcile the business ids with system ids and vice versa although it makes provision for the same.

4 LRA Query Model Specification

This specification was generated from the UML source model using the RTF Class Model Specification template (v1.1.0).

The section details the LRA Query Model along with a specification of the additional packages (see Section 4.1) generated to support the core query model (see Section 4.2).

Section 4.1 extends the EHR_EXTRACT package to include three main services required to enable LRA querying. These are the Available Shareable EHR, Shareable EHR System, and Shared Demographic Service.

Section 4.2 explains the various components of the LRA Query Model. The three main query types are Demographic Selection query, Population query, and Subject of Care Query. Each of these query types can have zero or more (0..*) Query Parameters, which may include among other data types, any constraint types such as SCT constraint expressions, literal constraints, regular expression constraints etc. A Demographic Selection query may return an Aggregated Result Set for a given population. Further details on the usage of the model are available in Sections 6 and 7, supported by examples in Section 8.

4.1 Package lra.technical

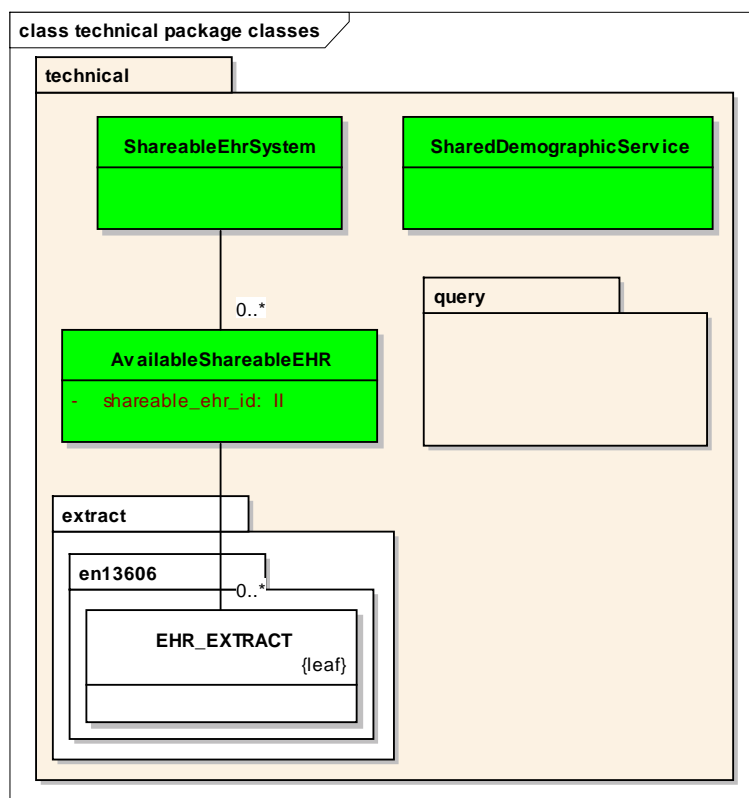


Figure 1: EHR and Demographics Extension Package

4.1.1 Class AvailableShareableEHR

Specialises:

Realises:

The available shareable EHR of a patient or service user. For the purpose of specifying LRA Query Definitions, each patient or service user known to the system has at the most one available shareable EHR.

Attributes

Attribute	Description
shareable_ehr_id : II [1..1]	The unique identifier of this shareable EHR, synonymous with the subject of care shareable id.

Relationships

Relationship Type	Source	Target	Description
Association	AvailableShareableEHR	EHR_EXTRACT 0..*	
Association	ShareableEhrSystem	AvailableShareableEHR 0..*	

4.1.2 Class ShareableEhrSystem

Specialises:

Realises:

A class representing a logical system capable of accessing the available shareable care record content (or parts thereof) of each of zero or more patients or service users for the purpose of specifying one or more LRA Query Definitions.

This class represents a logical abstraction for the purpose of specifying LRA Query Definitions and does not presume any particular **physical implementation** of the storage, organisation or method of accessing patients' or service users' care records.

4.1.2.1 Relationships

Relationship Type	Source	Target	Description
Association	ShareableEhrSystem	AvailableShareableEHR 0..*	

4.1.3 Class SharedDemographicService

Specialises:

Realises:

A class representing a logical system capable of accessing to the shareable content of the demographic records (or parts thereof) of patients or service users known to the system for the purpose of specifying one or more LRA Query Definitions.

For the purposes of query specification the demographic content to which the system has access is considered to include any necessary information content that conforms to the LRA Participations Reference Model (as defined in Reference Model package Ira::technical::participations).

4.2 Package Ira.technical.query

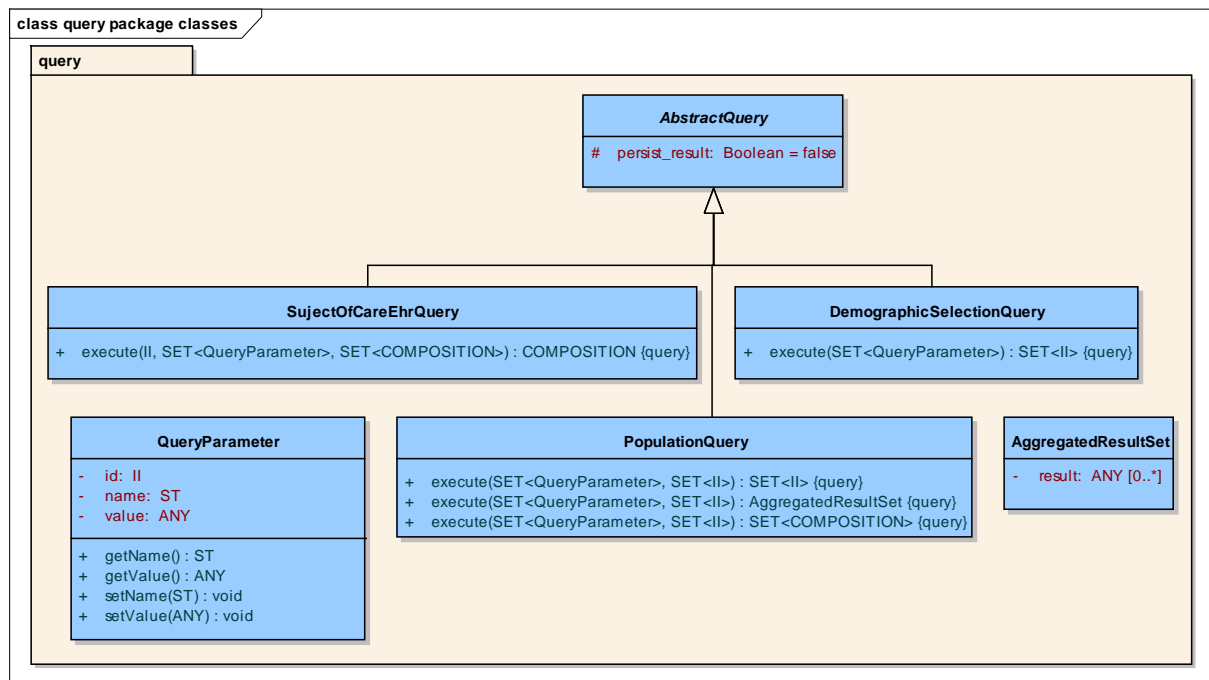


Figure 2: LRA Query Model

4.2.1 Abstract Class AbstractQuery

Specialises:

Realises:

The abstract generalisation of all LRA query classes.

4.2.1.1 Attributes

Attribute	Description
persist_result : Boolean [1..1]	True if the result of the query must be persisted, otherwise false.

4.2.1.2 Relationships

Relationship Type	Source	Target	Description
Generalization	SubjectOfCareEhrQuery	AbstractQuery	

Relationship Type	Source	Target	Description
Generalization	PopulationQuery	AbstractQuery	
Generalization	DemographicSelectionQuery	AbstractQuery	

4.2.2 Class AggregatedResultSet

Specialises:

Realises:

A container for one or more aggregated results of a population level query.

Properties to be defined.

4.2.2.1 Attributes

Attribute	Description
result : ANY [0..*]	A set of zero or more data values comprising result set.

4.2.3 Class DemographicSelectionQuery

Specialises: AbstractQuery

Realises:

This class defines an interface to a query method that provides for the specification of a demographic selection query. The query returns a set containing the shareable subject of care id of each subject of care with associated demographic information content that matches the query criteria specified by the supplied parameters.

4.2.3.1 Methods

Method	Description	Parameters
execute : SET<II>	Method returns a query result set containing the shareable id of each subject of care with associated demographic information content that matches the supplied query criteria.	params : SET<QueryParameter> The set of query parameters used by this query.

4.2.3.2 Relationships

Relationship Type	Source	Target	Description
Generalization			

Relationship Type	Source	Target	Description
	DemographicSelectionQuery	AbstractQuery	

4.2.4 Class PopulationQuery

Specialises: AbstractQuery

Realises:

This class defines interfaces to query methods that provide for the specification of a direct query, a derived operation or a direct inference on all or a subset of the EHRs of a population of patients or service users accessible to the shareable EHR system.

4.2.4.1 Methods

Method	Description	Parameters
execute : SET<II>	Method returns a query result set containing the shareable id of each subject of care with associated shareable EHR information content that matches the supplied query criteria. If the query range is NOT supplied then the query runs on the entire shareable EHR system.	params : SET<QueryParameter> The set of query parameters used by this query. qrange : SET<II> Optionally a set referencing one of more shareable EHR instances, each (ultimately) over which the query ranges.
execute : AggregatedResultSet	Method returns a set of one or more aggregated results of a population level query. If the query range is NOT supplied then the query runs on the entire shareable EHR system.	params : SET<QueryParameter> The set of query parameters used by this query. qrange : SET<II> Optionally a set referencing one of more shareable EHR instances, each (ultimately) over which the query ranges.
execute : SET<COMPOSITION>	Method returns a set of one or more COMPOSITIONs of individual subjects of care that form part of the population level query. If the query range is NOT supplied then the query runs on the entire shareable EHR system.	params : SET<QueryParameter> qrange : SET<II>

4.2.4.2 Relationships

Relationship Type	Source	Target	Description
Generalization	PopulationQuery	AbstractQuery	

Relationship Type	Source	Target	Description

4.2.5 Class QueryParameter

Specialises:

Realises:

A class whose instances are used to supply parameters to LRA query interfaces in the form of globally and uniquely identifiable name value pairs.

4.2.5.1 Attributes

Attribute	Description
id : II [1..1]	The globally unique identifier of this parameter instance.
name : ST [1..1]	The name of this parameter instance.
value : ANY [1..1]	The value of this parameter instance.

4.2.5.2 Methods

Method	Description	Parameters
getName : ST	Gets the name of this parameter instance.	
getValue : ANY	Gets the value of this parameter instance.	
setName : void	Sets the name of this parameter instance.	newVal : ST
setValue : void	Sets the value of this parameter instance.	newVal : ANY

4.2.6 Class SubjectOfCareEhrQuery

Specialises: AbstractQuery

Realises:

This class defines an interface to a query method that provides for the specification of a direct query, a derived operation or a direct inference on the whole or part of an EHR of a single patient or service user, using the query criteria specified by the supplied parameters.

4.2.6.1 Methods

Method	Description	Parameters
COMPOSITION : execute	Method returns a COMPOSITION containing the result of a direct query, derived operation or direct inference on the whole or part of an EHR of a single patient or service user that matches the supplied query criteria. If the query range is NOT supplied then the query runs on the entire available shared EHR of the patient or service user.	shareableEhrlid : II The shareable EHR id of the subject of care. params : SET<QueryParameter> The set of query parameters used by this query. qrange : SET<COMPOSITION> Optionally a set comprising one of more COMPOSITION instances, each (ultimately) derived from the available shareable EHR identified by the shareable EHR id, over which the query ranges.

4.2.6.2 Relationships

Relationship Type	Source	Target	Description
Generalization	SubjectOfCareEhrQuery	AbstractQuery	

5 LRA Query Grammar

The LRA query grammar specifies the logical representation of queries. This representation derives from the LRA Care Components Model based on the EN13606-1 EHR Reference Model [3]. The logical query representation may be physically implemented using any formal query language such as SQL, XQuery, or a computable UML graph amongst others [QUERY 39].

The queries are described such that they

- Specify the Reference Model class(es) containing the data required by the query
- The data will be represented using the Reference Model Data Types which conform to the ISO21090 datatypes [QUERY 61] [5].
- Support unit conversion of physical quantities expressed as UCUM units [QUERY 35].
- Specify the constraints on the relationships between Reference Model Classes. An unconstrained relationship is permitted (cartesian product) but must be explicitly stated. The constraints are applied only to those implicit in the Reference Model.
- Specify the attributes, or functions of attributes, to be included in the returned data
- Specify the constraints on attributes of those classes, including constraints on SNOMED CT expressions. The constraints on SNOMED CT expressions should conform to the specifications described in the LRA Technical Model Design [QUERY 51] [4].
- Support the following set operations on results [QUERY 43]
 - Intersection - identifying all the returned data record sets meeting multiple constraint sets
 - Union - identifying all the returned data record sets meeting either one constraint set or another (or indeed both)
 - Disjoint - identifying all the returned data record sets meeting one constraint set but not meeting another

- Support mathematical functions on attributes [QUERY 65]
- Support constraints on functions following aggregation [QUERY 65]
- Support constraints on relationships based on time [QUERY 57]
- Support the versioning, change control, and authorship and modifications [QUERY 7]

5.1 Logical and Mathematical Operations

5.1.1 Logical Operators

A pair of sub query trees or a pair of atomic queries can be combined together using the standard boolean operators into new query trees. Thus, boolean operators are always internal nodes in the query tree. The new query trees are presented as a single COMPOSITION for a Subject of Care EHR query, or a set of COMPOSITIONs, an AggregatedResultSet or SET<II> for a Population query, or a SET<II> for a Demographic Selection query. The Boolean operators that can be used to combine individual query results are shown in Table 2.

Notation	Operator	Description
&&	Binary AND	Set intersection of two or more atomic query results
	Binary OR	Set union of two or more atomic query results
!	Binary NOT	Set complement of two or more atomic query results

Table 2: Logical Operators to merge query results

5.1.2 Mathematical Operators

In addition to logically combining sub queries to generate a composite result, mathematical functions, as shown in Tables 3 and 4, can also be performed on atomic values as well as on Collections such as arrays, lists, and sets, amongst others.

The mathematical operators in Tables 3 and 4 must be able to handle the LRA numeric datatypes INT and REAL as well as PQ and MO, which add a requirement for these operators to handle units. LRA uses coded ordinals (CO) to meet specific analysis use cases, and must be permitted as parameters to mathematical functions. The Unified Code for Units of Measure (UCUM) is the core vocabulary for expressing quantities within the LRA. The Physical Quantity datatype PQ is constrained using UCUM, while PQV is reserved for physical quantities expressed using another code system. Comparison between values of type PQ expressed in different but commensurate units (such as 1 metre =100 centimetres) is supported in the ISO 21090 specification [7]. While automatic unit conversion for the purpose of value comparison will occur within the query using datatype semantics already defined, query output may be required in a specific unit (Taken from [2]).

Table 3 extends on the operations that can be performed on the ISO21090 data types [7] [QUERY 65]. Some of the standard operations already defined in the ISO21090 data types specification are addition, subtraction, division, absolute value, min value, and max value amongst others [7]. The list of operations is not exhaustive and can be extended if further operations are identified for querying [QUERY 66]. The data type returned by the operations shown in Tables 3 and 4 do not identify the UCUM units in which the data type is expressed as it is implicitly supported through the ISO21090 data type conformance. T denotes Type which is an abstract data type that needs to be specialised in order to be instantiated. A mathematical function may result in the same data type or might result in a conversion to another data type.

Function Name	Description
exp(x: T):T	Returns the value of e raised to the specified power.
log(x: T):T	Returns the natural (base e) logarithm of the argument.
log10(x: T):T	Returns the common (base 10) logarithm of the argument.

pow(x: T):T	Returns the first argument raised to the power of the second argument.
sqrt(x: T):T	Returns the square root of the argument.

Table 3: Extended mathematical functions on ISO21090 data types

Table 4 identifies a list of aggregate functions that can be performed on a Collection to return an appropriate data type [QUERY 65].

Function Name	Description
count(x: T, c: COLL (T)): INT	Returns the number of occurrences of object in a collection.
sum(c: COLL(T)):T	Returns the total of all the elements in a collection.
avg(c: COLL (T)):T	Returns the average (arithmetic mean) of the numerical elements in a collection
stdev(c: COLL (T)):REAL	Returns the standard deviation of the numerical elements in a collection.
variance(c: COLL (T)):REAL	Returns the variance of the numerical elements in a collection
median(c: COLL (T)):T	Returns the median of the numerical elements in a collection
mode(c: COLL (T)):T	Returns the most frequently occurring value in a collection.
max(c: COLL (T)):T	Return the largest value in the collection c.
min(c: COLL (T)):T	Return the smallest value in the collection c.
minus(c:COLL(T)):T	Return the smallest value in the collection c.

Table 4: Mathematical functions on Collections

6 Query Format

Each query will consist of a definition which will conform to the representation in Section 6.1. In addition, a query might be supported by one or more UML models such as a class or sequence diagram to facilitate technical as well as non-technical and clinical participants to understand and review the query [QUERY 14, 40].

A complex query can be broken down into smaller atomic queries to enable better understanding and readability. The query definition provides a human-readable format with sections indicating which queries are used by the query being defined (See 'uses queries' in Section 6.1) as well as the queries that use this query (See 'used by queries' in Section 6.1) [QUERY 5, 14]. Notes or comments may be added to the different components of a query i.e. query definition, query classes, query methods, and query attributes to clarify its usage or operation in the query as a whole [QUERY 6].

Each query may include metadata with information on the author, date of creation, status, review details, assumptions made and other details that will help in quality assurance and development [QUERY 11-13, 15]

6.1 Query Definition

A Query Definition Artefact provides the format in which a query will be described to satisfy the requirements specified in the knowledge space for a particular domain [QUERY 40, 41]. This formal query definition provides an interface between the knowledge and technical layers.

A query is described using the following parameters:

(i) Query Id – A unique identifier of the query [QUERY 2].
(ii) Query Name – A human-readable name assigned to the query, which conveys the general capability of the query [QUERY 4].
(iii) Query Rationale – The reasoning or justification for performing the query. E.g. the query might be required to determine the future care plan of a patient, or an imminent national pandemic.
(iv) Query Type – The type of a query is based on the following: <ul style="list-style-type: none"> (a) Reference Model type: The query type is classified according to its reference model class i.e. Subject of Care, Population, or Demographic Selection query [QUERY 18]. (b) Function type: In addition to the reference model type a query is also classifiable on the basis of the type of function performed, i.e. Direct, Derived Operation, or Direct Inference query. One or more functions may be applied to a single query, if required [QUERY 26, 48].
(v) Query Parameter – A single query can be decomposed into several types as mentioned in (iv) above. Each component of a query can comprise of its own parameter set [QUERY 8, 34] consisting of a: <ul style="list-style-type: none"> (a) Parameter name: A human-readable name assigned to the parameter. (b) Parameter value type: A parameter value as an ISO21090 data type [QUERY 34, 47, 61, 63]. This may include a SNOMED CT constraint expression, a literal constraint, or any other constraint type where appropriate [QUERY 28-33]. (c) Parameter description: A general description of the parameter. <p>A query will always be passed a parameter as a 'name-value' pair making it uniquely identifiable.</p>
(vi) Query Range – The query range differs for queries based on their reference model type [QUERY 44-46]: <ul style="list-style-type: none"> (a) For DemographicSelection Query – The default range of any query is the entire demographics service of the EHR system. However, the demographic range might be constrained by passing in specific subject(s) of care or service user information. (b) For SubjectofCareEHR Query – The default range of a subject of care query is the entire EHR of a single patient. However, a specific set of COMPOSITIONS might be queried instead of the entire EHR to constrain the search criteria. (c) For Population Query - The default range of a population query is the entire EHR of all subjects of care. However, the range of a query might be restricted to include only EHRs of a group of subjects of care. The query range might become even more restricted as query parameters and constraints are applied during execution of a query and/or its sub-queries.
(vii) Query Expression – A query expression is the formal representation of a query which is distinct from its implementation. In R3, the query will be expressed as a pseudo code (details in Section 7). A formally specified syntax for a logical (i.e. technology-independent and computable) query and inference expression language may be defined at a later stage [QUERY 39]. A query expression includes two main components: <ul style="list-style-type: none"> (a) Input and predicate clause: A query needs to express its input clause (e.g. FOR using XQuery or FROM using SQL) and a list of predicates (e.g. WHERE using XQuery or SQL) [QUERY 42].

(b) Result specification: A query result is specified based on the query type in the query reference (or static) model [QUERY 10, 27, 47]. The result data conforms to the ISO21090 datatypes [QUERY 61].

- 1) For DemographicSelection Query – Such queries return a query result set containing the shareable id of each subject of care with associated demographic information content that matches the supplied query criteria.
- 2) For SubjectofCareEHR Query – Such queries wrap the result in an instance of the LRA reference model i.e. a COMPOSITION. The path to an instance of the reference model is identified using the paths derived from the input and predicate clauses. The COMPOSITION result may be persisted within the patient EHR [QUERY 19]. If the result represents a derived operation or direct inference on existing data in the EHR, the new COMPOSITION must provide links to the existing data that was used through the rc_id attribute [QUERY 24] along with information on the attestation of this new data created using an instance of the ATTESTATION_INFO class in the LRA Reference Model [QUERY 25].
- 3) For Population Query – Such queries have two kinds of results that might be returned: (i) A set of shareable ids of each subject of care which fall within the population cohort satisfying the requirements of the query, or (ii) an aggregated result set which might be a single collective value or a cluster of collective values of the population cohort satisfying the query. The result set can be of ANY data type based on the data type of the relevant EHR data. Where values are held or derived in a different, commensurate unit the aggregated result will be recorded in a specified unit [QUERY 36].

Irrespective of the number and range of outputs requested by the query, a single composite COMPOSITION is returned. The data in the EHR is not modified at any point during the merger or simple display of the query result [QUERY 1]. E.g. a query which requires two separate values of height and weight of a patient will be combined together in a single COMPOSITION result set. Therefore, the COMPOSITION will have a Height ENTRY and a Weight ENTRY with a Height Observation ELEMENT value and a Weight Observation ELEMENT value respectively.

A query might not return any results or might fail in which case an appropriate 'null flavour' must be communicated to the user or system [QUERY 23, 55, 56, 64].

(viii) **Uses queries** – A query might use other queries as sub-queries to generate a result [QUERY 9, 16, 17].

(ix) **Used by queries** – A query might be used by other queries as a sub-query to help generate a result [QUERY 9, 16, 17].

7 Query Methods

The LRA Query Model defines a standard set of methods to be executed against a specific Reference Model query type (i.e. Demographic Selection, Subject of Care EHR, and Population).

- (a) DemographicSelection Query – This class has a single method which takes in any number of query criteria as input parameters and returns a query result set containing the shareable id of one or more subjects of care or service users.

Method signature: execute (SET<QueryParameter>): SET<II>

- (b) SubjectofCareEHR Query – This class has a single method that query on a specified set of parameters for a single shareable subject of care id. Optionally, the query may apply to one or more parts of the subject of care record i.e. COMPOSITION or the entire subject of care EHR (default range-see Section 6.1 (vii)(b)) [QUERY 48]. The method returns a single composite COMPOSITION with all the query results pertaining to a single subject of care. These methods provide for the specification of a direct, derived operation, or a direct inference query.

Method signature: execute (II, SET<QueryParameter>, SET<COMPOSITION>): COMPOSITION

- (c) Population Query - This class permits two kinds of query methods to be executed. A population query may return a set of shareable subject of care or service user identifiers which match a set of query criteria passed as an input parameter. The result is a subset of the set of shareable identifiers passed as the population range across which the query is to be executed.

Method Signature: execute (SET<QueryParameter>, SET<II>): SET<II>

Alternatively, the population query may return an aggregated result over a cohort of the population specified by the set of shareable identifiers in the input parameter. This aggregated result set could be a single value or a group of values as a result of a direct, derived operation or direct inference query.

Method Signature: execute (SET<QueryParameter>, SET<II>): AggregatedResultSet

Finally, the population query may alternatively return a set of one or more COMPOSITIONs of individual subjects of care that form part of the population level query. The result will be a cluster of individual COMPOSITIONs each representing a particular subject of care within the cohort.

Method Signature: execute (SET<QueryParameter>, SET<II>): SET<COMPOSITION>

In addition to the standard query methods which form part of the LRA Query Model, operational methods can also be applied to instances of the LRA Reference Model. Some basic operations have been defined in the query specification. However, any number of operational methods can be defined in order to facilitate querying, as deemed appropriate. In addition, the values of the record component attributes extracted using the operational methods can be further manipulated using the ISO21090 datatype-specific methods [QUERY 60, 62]. Operational methods have not been defined on the LRA Demographics Model as they are outside the scope of the current query specification (as stated earlier in Section 3.1).

The operational methods that have been added to the LRA Reference Model to facilitate querying are [QUERY 54]:

- (d) Operations on ELEMENTs – There are three main ELEMENT types i.e. BOUND_DATA_ELEMENT, UNBOUND_DATA_ELEMENT, and COMPONENT_RELATIONSHIP_ELEMENT. The two main ELEMENT types on which operations are defined are the BOUND_DATA_ELEMENT, and UNBOUND_DATA_ELEMENT. An instance of COMPONENT_RELATIONSHIP_ELEMENT specifies the relationship between two ELEMENTs and therefore is not queried directly.

(i) getObsTime(): IVL<TS> - This method returns the time interval when an observation is made by an instance of an ELEMENT (obs_time attribute). An instance of an ELEMENT can be either a kind of BOUND_DATA_ELEMENT, or UNBOUND_DATA_ELEMENT. Further operations may be performed on the obs time to determine, for instance, the obs start date and time, obs end date and time, and/or the duration of the observation [QUERY 57, 59].

(ii) `getEventTime(): IVL<TS>` - This method returns the time interval when an event occurred. If the `ELEMENT.obs_time` is null then the `COMPOSITION.session_time` is returned instead [QUERY 57, 59]. See Section 4.2.2 of the LRA Technical Model Design document [4] for details on the event time.

(iii) `getMeaning(): CD.CV.SCT` - The method returns the meaning of an instance of a kind of `BOUND_DATA_ELEMENT`. The meaning is defined as a UUID, or a SNOMED CT instance identifier, or a SNOMED CT constraint expression [QUERY 49-52]. The meaning is queried to determine whether the instance is relevant to a particular query by conforming to the query's semantic constraint passed as an input parameter.

(iv) `getValue(): ANY` – This method is applicable to only three kinds of `ELEMENTs` viz. `UNBOUND_DATA_ELEMENT`, `PROPERTY_OBSERVATION_ELEMENT`, and `MATERIAL_ENTITY_ELEMENT`. All other `ELEMENT` types have their value type set to 'null' and therefore cannot be queried.

1. For `UNBOUND_DATA_ELEMENT` instance - `getValue(): ANY` - This method retrieves ANY value of an instance of an `UNBOUND_DATA_ELEMENT` from the value attribute.
2. For `PROPERTY_OBSERVATION_ELEMENT` instance – `getValue(): PQ` - This method retrieves a physical quantity (PQ) value of an instance of a property observation. Only that observation value is retrieved which satisfies the query's semantic constraint using the `getMeaning()` method.
3. For `MATERIAL_ENTITY_ELEMENT` instance – `getValue(): QTY` - This method retrieves a quantity (QTY) value of an instance of the material. Only that material instance value is retrieved which satisfies the query's semantic constraint using the `getMeaning()` method.

(e) Operations on `COMPOSITIONs` – All records are contained within `COMPOSITIONs`. Each instance of a `COMPOSITION` records the date and time or interval during which a clinical encounter or documentation session occurred. A single operation is defined on a `COMPOSITION`.

(i) `getSessionTime(): IVL<TS>` - This method returns the time interval when a particular session occurred (`session_time` attribute). Further operations may be performed on the time interval to obtain the session start and end time, duration of a session, amongst others [QUERY 57, 59].

7.1 Computable Query Expression

A Computable Query Expression is a formal expression specifying the retrieval criteria and result set of a query as specified in section 6.1 (see Query Definition).

A formal query expression consists of a subject and a list of predicates. In R3, a query expression will be represented in pseudo code using the key clauses of an XQuery FLWOR expression. The key clauses include `FOR`, `LET`, `WHERE`, `ORDER BY`, and `RESULT (FLWOR)`. However, the pseudo code expressions will be converted to a formal representation such as OCL, SQL, XQuery, or UML graphs at a later stage so that implementers of the LRA queries can easily transform the expressions to computable query syntax of their choice. NOTE: LRA does not propose the use of any specific computable query syntax.

A typical query expression used in R3 is shown below:

<code>FOR</code> < optional: input sequence i.e. the subject query range>
<code>LET</code> < optional: declare a variable and assign it a value>
<code>WHERE</code> < optional: list of predicates i.e. conditions to filter items of interest>
<code>ORDER BY</code> <optional: presents results in a preferred order> [QUERY 21, 22]

RETURN <list of items included in the result>

7.2 Temporal Queries

Section 4.2 in the LRA Technical Model Design document [4] details the LRA approach to time-based queries. The temporal information of interest to querying in terms of complexity is the temporal expressions on clinical content. The Design document [4] states that the attributes used to represent the temporality of care record information are:

- COMPOSITION session_time and ELEMENT obs_time (which together can be used to infer the event time) (see Section 7(d) for methods on session_time and obs_time) [QUERY 57, 59].
- The temporal context and finding or procedure context of the meaning attribute value for representing contemporary and retrospective information. The default constraint for the temporal context is: 408731000 | temporal context | = < 410510008 | temporal context value |.
 - (i) Contemporary information is indicated by a temporal context value that conforms to the following constraint: 408731000 | temporal context | = ((<< 410512000 | current or specified |) AND (! < 410513005 | past |)).
 - (ii) Retrospective information is indicated by a temporal context value that conforms to the following constraint: 408731000 | temporal context | = ((<< 410511007 | current or past |) AND (! < 15240007 | current |));
- Finding or procedure context and the UNBOUND_DATA_ELEMENT FutureEventTime for representing prospective events. A prospect (e.g. expectation, goal, risk, intended event, etc) is represented as an observation or activity with an appropriate finding or procedure context value.
 - (i) For observations the prospect of a finding is indicated using one of the *finding context* values (or a subtype thereof) specified by the following constraint: 408729009 | finding context | = ((<< 410519009 | at risk |) OR (<< 410517006 | expectation |) OR (<< 410518001 | goal |))
 - (ii) For activities the *procedure context* of the meaning attribute is used to indicate the degree of completion, or status of the associated procedure. Activities that have yet to occur are indicated using an appropriate subtype of *pre-starting action status* for the procedure context value as defined by the following constraint: 408730004 | procedure context | = ((< 410522006 | pre-starting action status |) OR (< 410537005 | action status unknown |))
- UNBOUND_DATA_ELEMENT ValidPeriod for representing the period of validity of a prospect or entity. The period of validity such as the start and end date of a plan or the expiration date of a drug is represented by semantically linking an instance of UNBOUND_DATA_ELEMENT ValidPeriod to the observation, activity or material entity.
- ELEMENT multiplicity and obs_time used for representing time series. A series of individually timed events is represented by specifying an activity or observation element with an upper bound multiplicity of greater than 1 and using the obs_time of each recorded instance to record the time of the particular event that it represents.

Each of the representations of time can be queried using the getSessionTime(), getEventTime(), or getObsTime() operations or by querying an instance of the meaning constraint using a suitable query SCT constraint expression as an input parameter.

8 Query Examples

The section is an illustration of the query specification detailed in the document and serves as a proof of concept. There are three main examples which have been broken down into several individual sub-queries. These are:

(i) **Patient-level hypothyroidism medication example** (see Appendix A.1 for a diagrammatic representation of the example). Section 8.2.1.1 defines the main query named 'pat_Med-based_diagnosis_query' which calls a number of sub-queries to retrieve the required result.

```

Sec 8.2.1.1 (pat_Med-based_diagnosis_query)
    →usesQueries →
    Sec 8.1.1.1 (pat_shareable_id_query)
    Sec 8.2.1.2 (pat_med_query)
        →usesQueries →
        Sec 8.2.1.3 (pat_diag_query)
  
```

The patient-level hypothyroidism example can be extended to a population-level example in which case the main query will become the 'population_hypothyroidism_query' in Section 8.3.1.1. The traversal route will then be:

```

Sec 8.3.1.1 (population_hypothyroidism_query)
    →usesQueries →
    Sec 8.2.1.1 (pat_Med-based_diagnosis_query)
        →usesQueries →
        Sec 8.1.1.1 (pat_shareable_id_query)
        Sec 8.2.1.2 (pat_med_query)
            → usesQueries →
            Sec 8.2.1.3 (pat_diag_query)
  
```

(ii) **Population-level obesity example** (see Appendix A.2 for a diagrammatic representation of the example). Section 8.3.2.1 defines the main query named 'population_obesity_query' which calls a number of sub-queries to retrieve the required result for the given population.

```

Sec 8.3.2.1 (population_obesity_query)
    →usesQueries →
    Sec 8.1.1.2 (pct_Dob_Query)
    Sec 8.2.3.1 (obesity_finding_query)
        → usesQueries →
        Sec 8.2.2.1 (bmi_result_query)
            → usesQueries →
            Sec 8.2.1.4 (height_Query)
            Sec 8.2.1.5 (weight_Query)
  
```

(iii) **Haemodialysis session information example** which extends from the knowledge layer (see Appendix A.3 for a diagrammatic representation of the example). This is an end-to-end example and demonstrates the flow of query requirements from the knowledge layer of the LRA to its realisation in the LRA technical models. Figure 3 diagrammatically presents the haemodialysis example from the technical layer with traceability to the knowledge layer.

Due to the large size of the haemodialysis example, a section of it has been selected for illustrating the query definitions in the following sub-sections. The traversal path of the chosen section of the composite whole is shown below.

[Sec 8.3.1.2](#) (Pop_HD_session_info_query)

→usesQueries →

[Sec 8.1.1.3](#) (Unit_patient_list_query)

[Sec 8.2.1.6](#) (patient_HD_session_info_query)

→ usesQueries →

[Sec 8.2.2.2](#) (pat_session_URR_query)

→ usesQueries →

[Sec 8.2.1.7](#) (pre_HD_urea_query)

[Sec 8.2.1.8](#) (post_HD_urea_query)

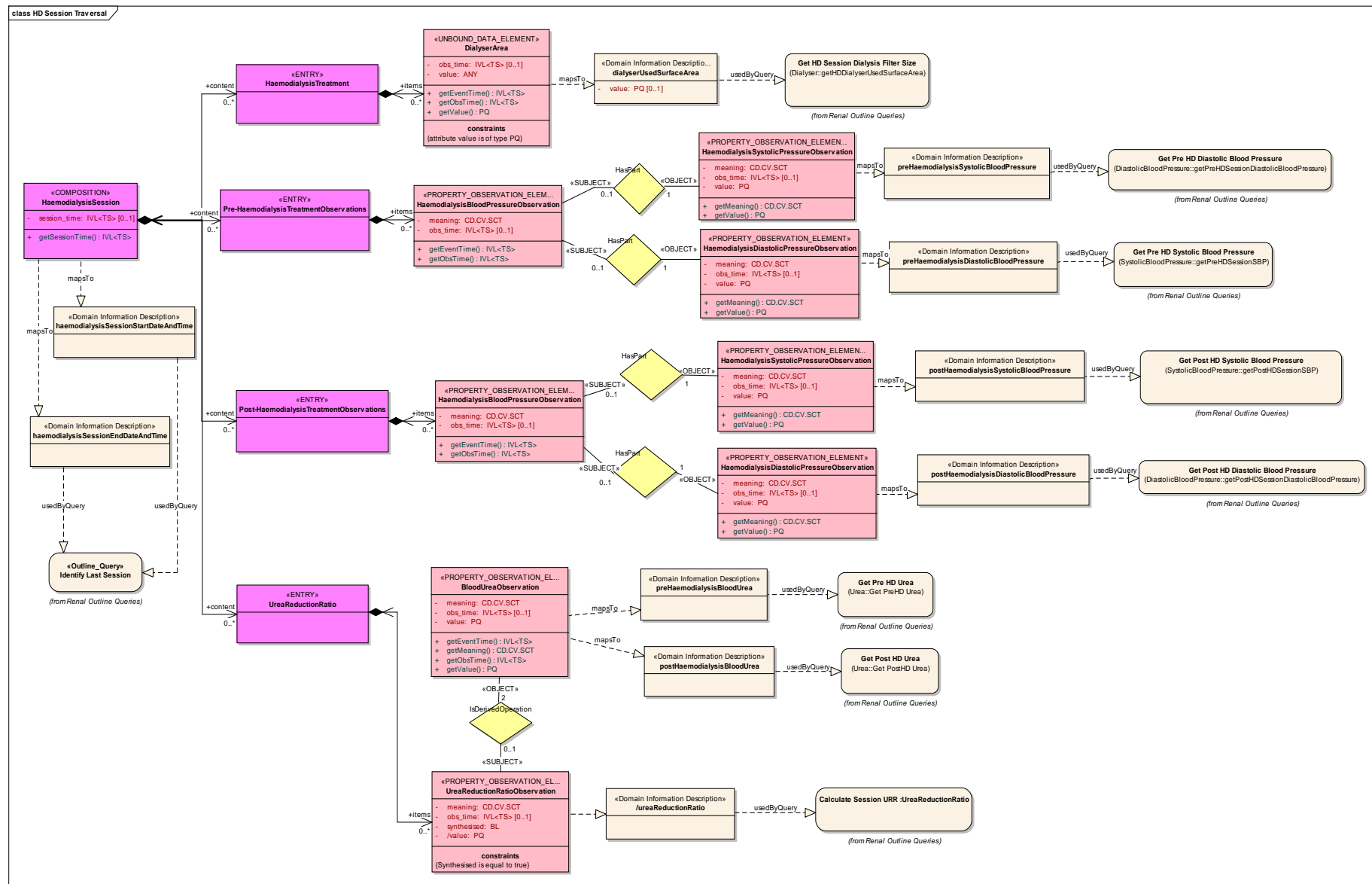


Figure 3: Traversal of Haemodialysis session example from technical to knowledge layer

8.1 Demographic Selection Queries

8.1.1 Direct Queries

8.1.1.1 Shareable Patient Identifier

Query ID - {10B39D79-6D04-4276-A910-C1C410FC5C97}

Query Name - pat_shareable_id_query

Description – Return a shareable unique system identifier of a patient given the patient's business identifier.

Rationale – This query is required to return a shareable subject of care id of a patient with associated demographic information content that matches the supplied patient business identifier. This helps in providing a standard approach to querying patient data.

Query Type –

Reference Model Type: Demographic Selection Query.

Function Type: Direct Query.

Query Parameter – The initial parameters are

Name	Value Type (ISO21090 data types)	Value (optional)	Description (optional)
Pat.id	II	XYZ12345	Unique identifier of a patient such as an NHS number.

Query Range – The virtual shared demographic service that holds all demographic information about the patient.

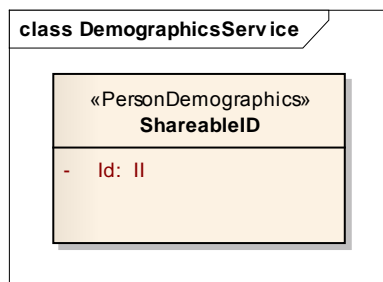
Query Expression –

FOR \$x in SharedDemographicService/RolePerson (: A sample of how the demographics model might be queried :)

WHERE \$x/@patId = XYZ12345

RESULT {\$x/id}

(: Returns a shareable system identifier of the patient given their business identifier :)

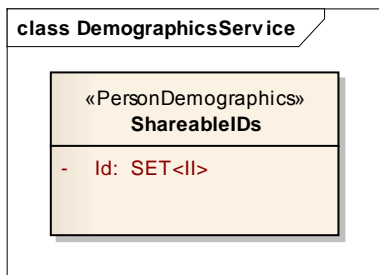


Uses queries – None
Used by queries – pat_Med-based_diagnosis_query, pat_med_query, pat_diag_query

8.1.1.2 Set of shareable organisation-specific patient identifiers

Query ID: {454C9242-9D89-4eec-A669-1EA5665ACBC0}			
Query Name: pct_Dob_Query			
Description – Return a set of unique system identifiers of all children who belong to a particular PCT.			
Rationale – This query is required to return a set containing the shareable subject of care id of each child with associated demographic information content that matches the supplied PCT organisation code.			
Query Type –			
Reference Model Type: Demographic Selection Query.			
Function Type: Direct query.			
Query Parameter – The initial parameters are			
Name	Value Type (ISO21090 data types)	Value (optional)	Description (optional)
Org.code	II	ABC123	The unique identifier of the organisation to which a patient belongs or has some association. ABC123 is the code of the PCT responsible for schools in an area.
person.dob	IVL<TS>	2005..2006	Age Range (which is the target population of 4-5 year olds)
Query Range – The virtual shared demographic service that holds all demographic information about the PCT and the children registered with that PCT.			
Query Expression –			
FOR \$x in SharedDemographicService/RolePerson (: A sample of how the demographics model might be queried :)			
FOR \$y in SharedDemographicService/RoleOrganisation			
WHERE \$y/@orgCode = ABC123 and \$x/@dobRange = <2005..2006>			
RESULT {\$x/id}			
(: Returns set of system identifiers of all people who are registered with the specified PCT and were			

born in either 2005 or 2006 :)



Uses queries – None

Used by queries - population_obesity_query, height_and_weight_Query

8.1.1.3 Dialysis Unit Patient List Query

Query ID: {184069A9-17C8-4a49-9E5D-BA1EF43D41DC}

Query Name: Unit_patient_list_query

Description – Returns a set of unique system identifiers of all patients who attended the given renal dialysis unit.

Rationale – This query is required to return a set containing the shareable subject of care id of each patient with associated demographic information content that matches the given renal dialysis unit.

Query Type –

Reference Model Type: Demographic Selection Query.

Function Type: Direct query.

Query Parameter – The initial parameters are

Name	Value Type (ISO21090 data types)	Value (optional)	Description (optional)
Dialysis.unit	II	RU09990	The unique identifier of the unit for which haemodialysis session information of all patients is required.

Query Range – The virtual shared demographic service that holds all demographic information about the renal dialysis unit.

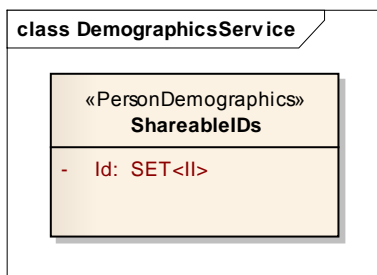
Query Expression –

FOR \$x in SharedDemographicService/RoleOrganisation (: A sample of how the demographics model might be queried :)

WHERE \$x/@orgCode = RU09990

RESULT {\$x/id}

(: Returns set of system identifiers of all people who have attended a haemodialysis session in the given renal dialysis unit :)



Uses queries – None

Used by queries – Pop_HD_session_info_query

8.2 Subject of Care EHR Queries

8.2.1 Direct Queries

8.2.1.1 Medication-based Patient Diagnosis Query

Query ID - {4603CFDE-2C94-42b7-9A02-54D74056ADE9}

Query Name - pat_Med-based_diagnosis_query

Description - Produce a report to indicate whether patient XYZ has hypothyroidism provided they are on a repeat prescription of thyroxine.

[Adapted from THYROID 1 section of Quality and Outcomes Framework Guidance – Aug 2004]

Rationale - A register is a prerequisite for monitoring patients with hypothyroidism. Many patients will have been diagnosed at some time in the past and the details of the diagnostic criteria may not be available. For this reason a register should be maintained for those patients taking thyroxine with a recorded diagnosis of hypothyroidism. The most effective method for identifying a patient would be a computer search for repeat prescribing of thyroxine with a subsequent check of the records to confirm the clinical diagnosis.

Query Type –

Reference Model Type: Patient Query.

Function Type: Direct Query.

Query Parameter – The initial parameters are

Name	Value Type (ISO21090 data types)	Value (optional)	Description (optional)
Pat.id	II	XYZ12345	Unique identifier of a patient such as an NHS number.
Med.type	ST	Thyroxine	The medication that is being searched in the patient EHR.
Diag.type	ST	Hypothyroidism	The name of the diagnosis that is being searched in the patient EHR
isRepeat	BL	True	To check whether the medication is on a repeat prescription. By default the value is set to true.

Query Range – The entire EHR of the specified patient.

Query Expression –

FOR \$x in SubjectOfCareEHR/COMPOSITION/Hypo_Diagnosis
WHERE Med_COMPOSITION/Thyroxine_SupplyPlan = exists
RESULT \$x

(: Return a COMPOSITION of an instance of the hypothyroidism diagnosis if there exists a thyroxine medication supply plan :)

class Diagnosis

```

classDiagram
    class HypothyroidismDiagnosis {
        <<COMPOSITION>>
        - contribution_id: II [0..1]
        - name: SC {readOnly}
        - rc_id: II
        - session_time: IVL<TS> [0..1]
        - synthesised: BL
        + getSessionTime() : IVL<TS>
    }
    class Hypothyroidism {
        <<ENTRY>>
        - name: SC {readOnly}
        - rc_id: II
        - synthesised: BL
    }
    class HypothyroidismFinding {
        <<FINDING_OBSERVATION_ELEMENT>>
        - meaning: CD.CV.SCT
        - obs_time: IVL<TS> [0..1]
        - rc_id: II
        - synthesised: BL
        + getEventTime() : IVL<TS>
        + getMeaning() : CD.CV.SCT
        + getObsTime() : IVL<TS>
    }
    HypothyroidismDiagnosis "0..*" --> "0..*" Hypothyroidism : +content
    Hypothyroidism "0..*" --> "0..*" HypothyroidismFinding : +items

```


Uses queries – pat_shareable_id_query, pat_med_query, pat_diag_query

Used by Queries - None

8.2.1.2 Patient Repeat Medication Query

Query ID - {DAFC1978-4D3F-4c89-8EB9-50AF91C17E6E}

Query Name - pat_med_query

Description - Check whether a patient is currently on a repeat thyroxine medication.

Rationale – The query is executed to determine whether a patient has hypothyroidism based on the presence of one or more records indicating that the patient is 'currently' on a 'repeat' 'thyroxine' medication. If yes then a separate query will be executed to check whether the patient has ever been diagnosed (current and/or past records) with hypothyroidism.

Query Type –

Reference Model Type: Subject of Care EHR Query.

Function Type: Direct Query.

Query Parameter – The initial parameters are

Name	Value Type (ISO21090 data types)	Value (optional)	Description (optional)
Shareable.pat.id	II		Shareable system identifier of a patient.
Mat_activity.thyr	ST.CONSTR.SEMANTIC	Expression for < 38076006 thyroxine product	The semantic meaning of the thyroxine medication being searched in the patient's Medication Compositions.
isRepeat	BL	True	To check whether the medication is on a repeat prescription. By default the value is set to true.

Query Range – All COMPOSITIONS of the patient's EHR which relate to medication (Med_Compositions).

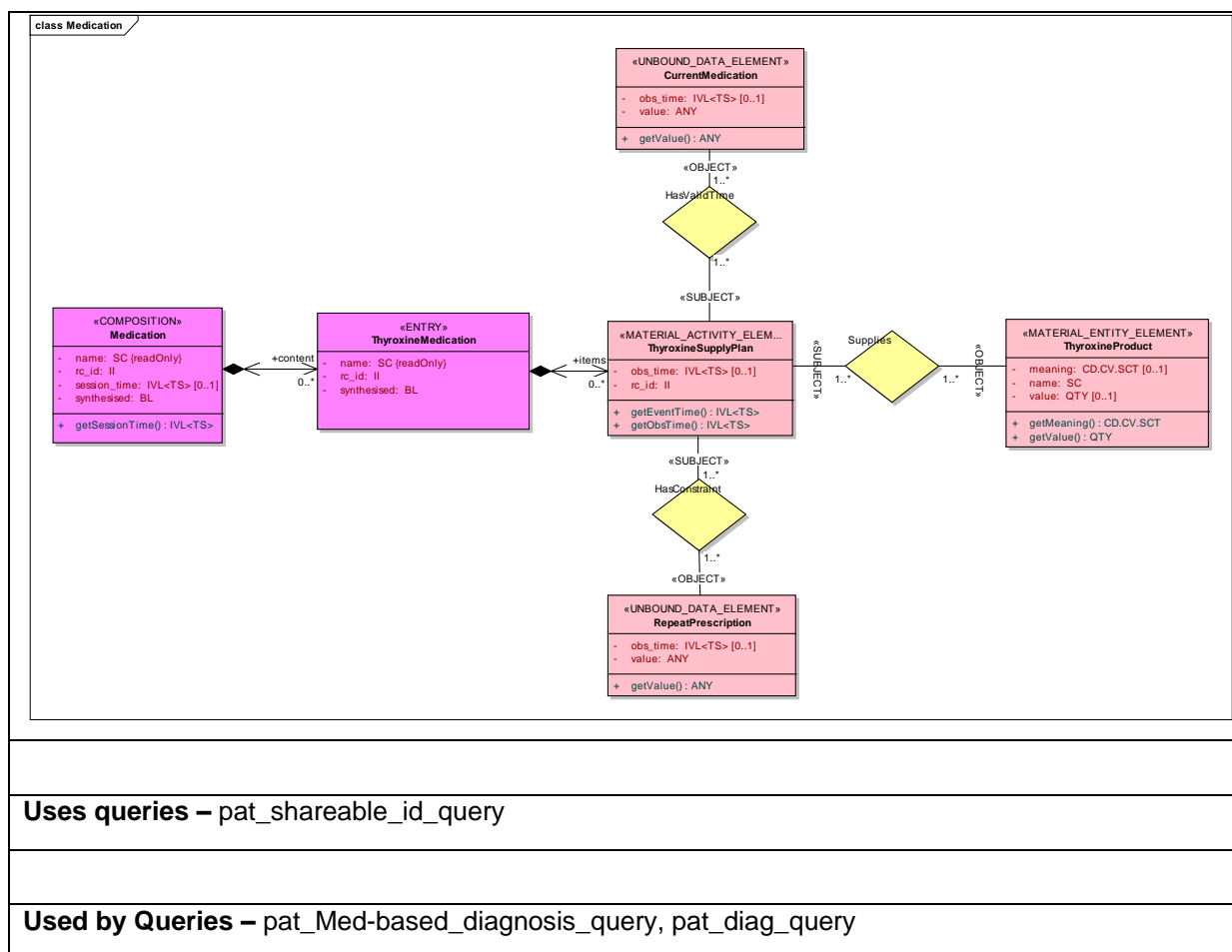
Query Expression –

FOR \$x in Med_COMPOSITION/Thyroxine_SupplyPlan.consumes().Thyroxine_MedItem
 FOR \$y in Med_COMPOSITION/Thyroxine_SupplyPlan.hasConstraint().Prescrip_RepeatNumber
 FOR \$z in Med_COMPOSITION/Thyroxine_SupplyPlan.hasValidTime().Prescrip_ValidityPeriod

RESULT if ((\$x/@meaning=< 38076006 | thyroxine product|) AND (\$y/@value > 1) AND (\$z/@value.highValue > currentTime()))

then Med_COMPOSITION/Thyroxine_SupplyPlan

(: Return the instance of the medication supply plan which contains the information about the repeat prescription medication thyroxine :)



8.2.1.3 Patient Diagnosis Query

Query ID - {E70C62AE-5E16-436f-879E-69BD4791FF7C}

Query Name - pat_diag_query

Description - Check whether a patient has ever been diagnosed with hypothyroidism based on their current consumption of the thyroxine medication.

Rationale – This query checks whether the patient has ever been diagnosed with hypothyroidism due to which the thyroxine medication has been prescribed. All current and/or past records of the patient are checked for a finding of the hypothyroidism diagnosis (direct query). If a record of hypothyroidism is not found in the patient EHR although they are currently on thyroxine then a new record is created with the diagnosis which may be persisted in the patient EHR (direct inference query) [QUERY 3].

Query Type –

Reference Model Type: Subject of Care EHR Query.

Function Type: Direct Query, (optionally) Direct Inference Query.

Query Parameter – The initial parameters are

Name	Value Type (ISO21090 data types)	Value (optional)	Description (optional)
Shareable.pat.id	II		Shareable system identifier of a patient.
Findg_obs.hypo	ST.CONSTR.SEMANTIC	Expression for < 40930008 hypothyroidism 	The semantic meaning of the hypothyroidism diagnosis being searched in the patient's EHR.
Diag.type	ST	Hypothyroidism	The name of the diagnosis being checked (could be used to locate appropriate diagnosis RECORD ARTEFACT ELEMENTs).

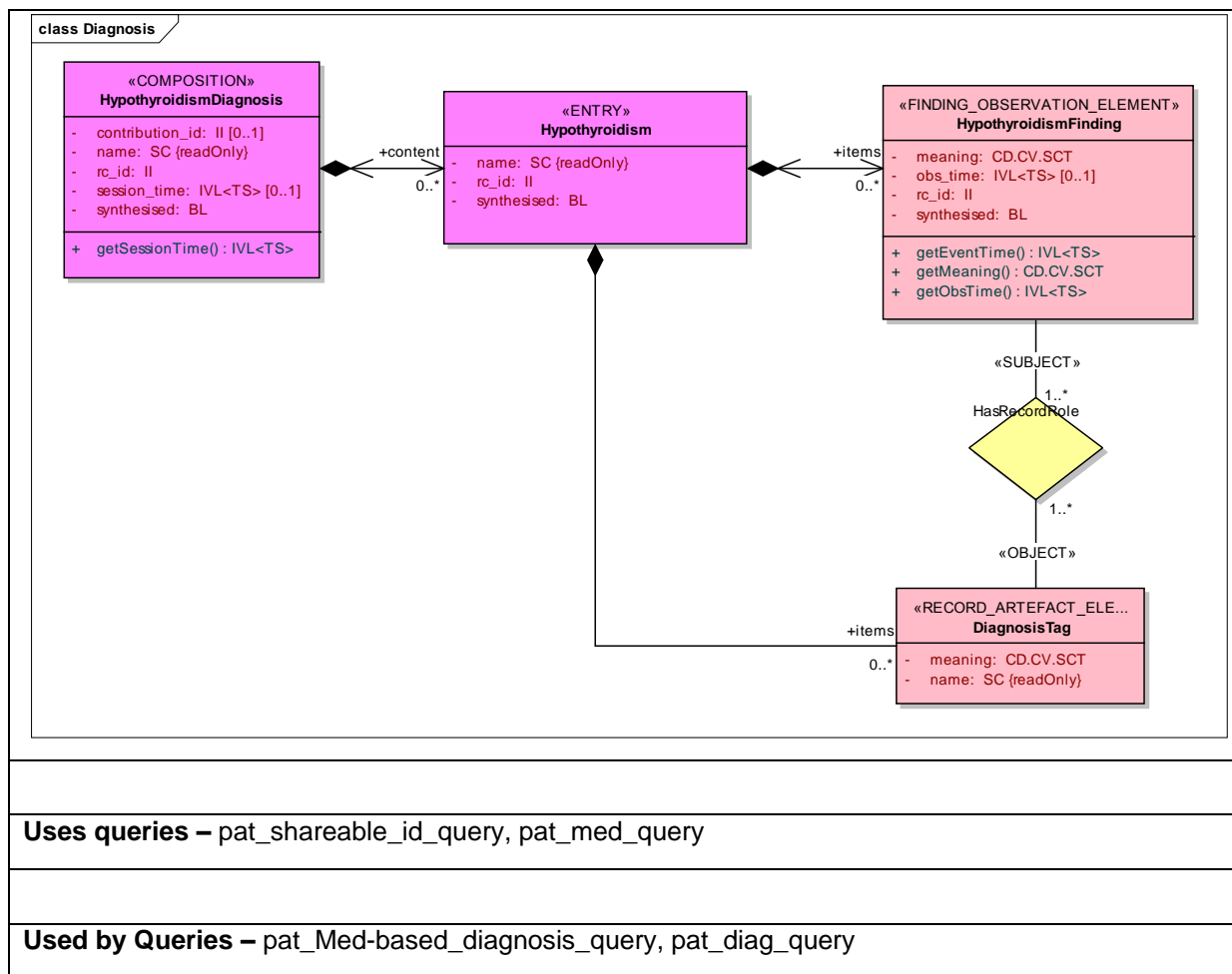
Query Range – The entire EHR of the patient.

Query Expression –

FOR \$x in SubjectOfCareEHR/COMPOSITION/Hypo_Diagnosis.hasItem().Hypo_DiagnosisName
 FOR \$y in SubjectOfCareEHR/COMPOSITION/Hypo_Diagnosis.hasItem().Hypo_DiagnosisTag

RESULT if ((\$x/@meaning = < 40930008 | hypothyroidism |) AND (\$y/@meaning = Hypothyroidism))
 then SubjectOfCareEHR/COMPOSITION/Hypo_Diagnosis

(: Return the instance of the hypothyroidism diagnosis which contains the information about the diagnosis made or created (if a record does not already exist) :)



8.2.1.4 Person Height Query

Query ID: {DED12DEE-3827-48d3-969D-19D9E41AF706}

Query Name: height_Query

Description - Return the height of all children given their unique system identifier. This query is run iteratively for each child (or subject of care) id returned.

Rationale - The query is executed to return the height value of all children within a PCT if the instance of height satisfy the given SNOMED CT (SCT) constraint expression.

Query Type –

Reference Model Type: SubjectofCareEHR Query.

Function Type: Direct Query

Query Parameter – The initial parameters are

Name	Value Type (ISO21090 data types)	Value (optional)	Description (optional)
Ehr.id.set	II		The shareable set of identifiers of children who belong to a particular PCT.
sct_height_constr	ST.CONSTR. SEMANTIC		The SCT constraint that defines the semantics of the instance of height to be queried

Query Range – The EHR which belongs to the child system id being queried iteratively.**Query Expression –**

FOR \$z in (ehr.id.set)

LET \$x := SubjectofCareEHR[@id=\$z]/COMPOSITION/BodyHeight_Entry/BodyHeightObs_Element

(: The parts of the EHR extract that needs to be queried :)

WHERE \$x/@meaning = (50373000|body height measure|)

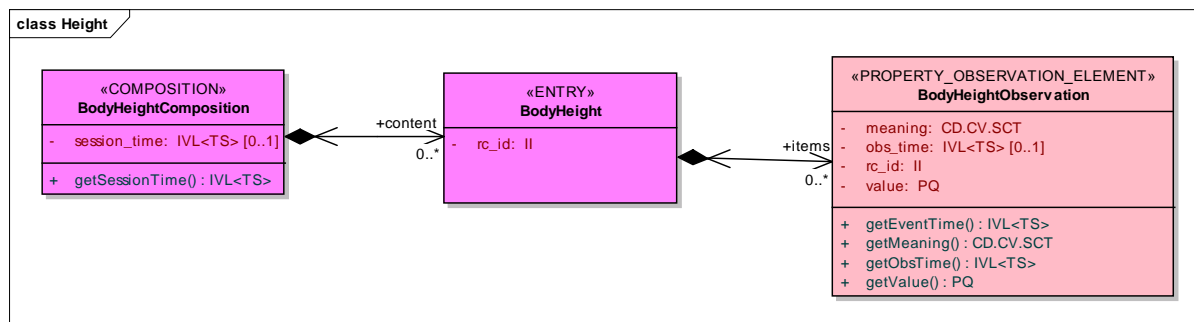
(: Provide the height SCT constraint expressions as the predicate. The complete constraint expression is available in the Notes section at the end of this query definition :)

LET \$a = \$x/value (: Store the height value in an interim variable :)

RESULT

SubjectofCareEHR/COMPOSITION/BodyHeight_Entry/BodyHeightResult_Element/@value=\$a,

(: Create a new composition with the result of the person body height :)

**Uses queries –** pct_dob_query**Used by queries –** population_obesity_query, bmi_result_query**Notes -**

The complete sct_height_constr is

243796009 | situation with explicit context | :
 { 246090004 | associated finding | = 50373000 | body height measure |

, 408729009 | finding context | = 410515003 | known present |
 , 408731000 | temporal context | = 410586007 | specified time |
 , 408732007 | subject relationship context | = < 125676002 | person |

8.2.1.5 Person Weight Query

Query ID: {38C82925-DA71-43b2-9B5B-1681AACA9A59}

Query Name: weight_Query

Description - Return the weight of all children given their unique system identifier. This query is run iteratively for each child (or subject of care) id returned.

Rationale - The query is executed to return the weight value of all children within a PCT if the instances of weight satisfy the given SNOMED CT (SCT) constraint expression.

Query Type –

Reference Model Type: SubjectofCareEHR Query.

Function Type: Direct Query.

Query Parameter – The initial parameters are

Name	Value Type (ISO21090 data types)	Value (optional)	Description (optional)
Ehr.id.set	II		The shareable set of identifiers of children who belong to a particular PCT.
sct_weight_constr	ST.CONSTR. SEMANTIC		The SCT constraint that defines the semantics of the instance of weight to be queried

Query Range – The EHR which belongs to the child system id being queried iteratively.

Query Expression –

FOR \$z in (ehr.id.set)

FOR \$x in SubjectofCareEHR[@id=\$z]/COMPOSITION/BodyWeight_Entry/BodyWeightObs_Element
 (: The parts of the EHR extract that needs to be queried :)

WHERE \$x/@meaning = (363808001|body weight measure|)

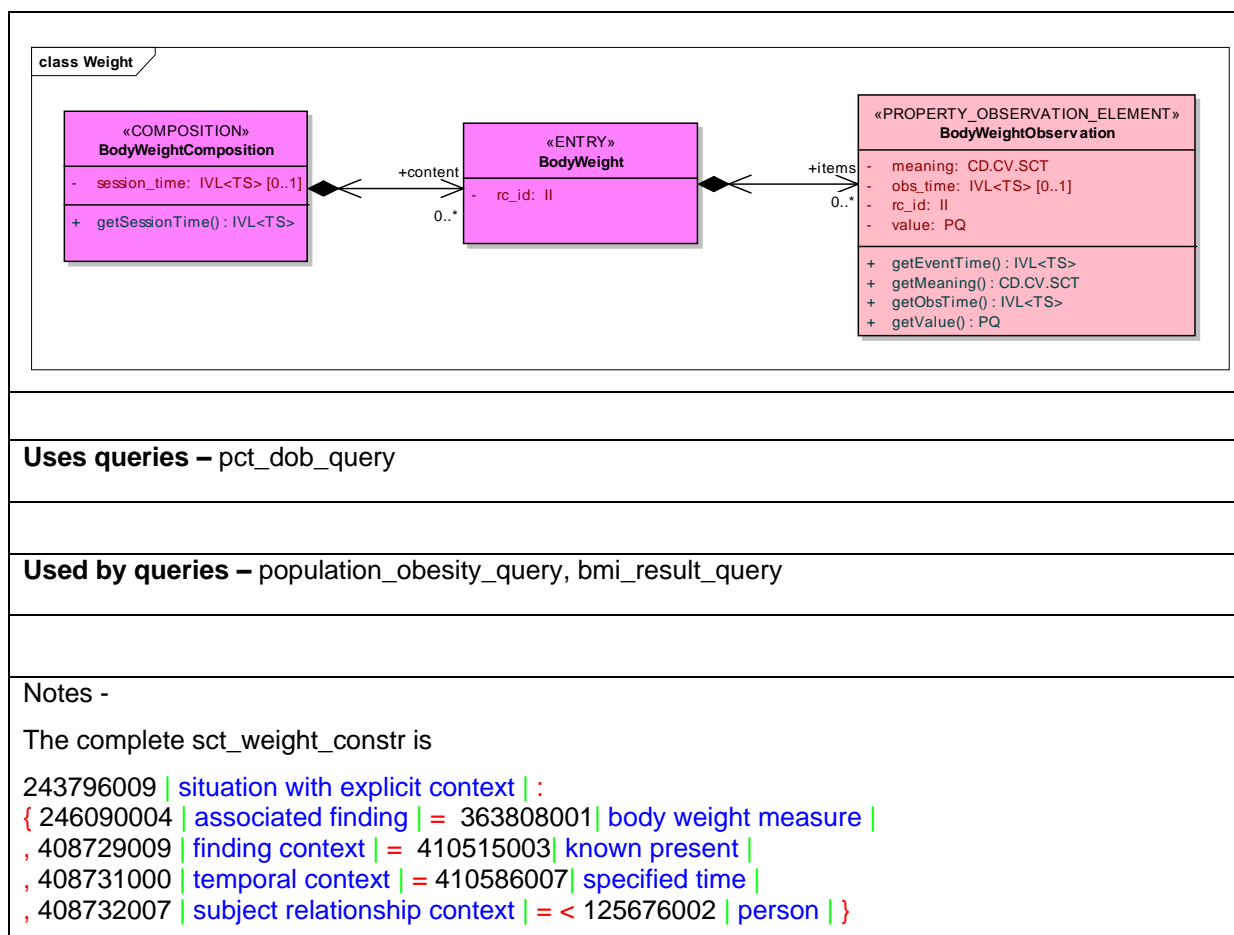
(: Provide the weight SCT constraint expression as the predicate. The complete constraint expression is available in the Notes section at the end of this query definition :)

LET \$a = \$x/value (: Store the weight value in an interim variable :)

RESULT

SubjectofCareEHR/COMPOSITION/BodyWeight_Entry/BodyWeightResult_Element/@value=\$a

(: Create a new composition with the result of the person body weight :)



8.2.1.6 Patient HD Session Information Query

Query ID: {525BCCC6-242E-446b-9DC9-5579839A92F3}			
Query Name: patient_HD_session_info_query			
Description – Returns the haemodialysis session information for a particular patient attending a dialysis unit.			
Rationale – The query retrieves the pre and post diastolic and systolic blood pressure, dialyser surface area, and the session URR for a patient attending a dialysis unit. NOTE: For the purpose of demonstration, only the session URR will be returned due to the large size of the query.			
Query Type –			
Reference Model Type: SubjectofCareEHR Query.			
Function Type: Direct Query.			
Query Parameter – The initial parameters are			
Name	Value Type	Value	Description (optional)

	(ISO21090 data types)	(optional)	
Start.date.time	TS.DateTime	201003170900	The start date and time of the haemodialysis session.
End.date.time	TS.DateTime	201003171800	The haemodialysis session end date and time.
Ehr.id	II		The shareable identifier of the patient who has had haemodialysis in a particular renal dialysis unit.
Query Range – All the COMPOSITIONs containing individual units of information about the patient's haemodialysis session. In this illustration this includes the COMPOSITION about the patient's session URR.			
Query Expression – FOR \$z in (Ehr.id) FOR \$x in SubjectofCareEHR[@id=\$z]/URR_Composition/UreaReductionRatio/UreaReductionRatioObservation (: The parts of the EHR extract that needs to be queried :) RESULT SubjectofCareEHR/HaemodialysisSession/UreaReductionRatio/UreaReductionRatioObservation (: Create a new composition with the result of the session URR :)			
<pre> classDiagram class PatientHDSession { class «COMPOSITION» HaemodialysisSession { - session_time: IVL<TS> [0..1] + getSessionTime() : IVL<TS> } class «ENTRY» UreaReductionRatio class «PROPERTY_OBSERVATION_ELEMENT» UreaReductionRatioObservation { - meaning: CD.CV.SCT - obs_time: IVL<TS> [0..1] - synthesised: BL - /value: PQ constraints { Synthesised is equal to true } } HaemodialysisSession "0..*" --> "1" UreaReductionRatio : +content UreaReductionRatio "0..*" --> "0..*" UreaReductionRatioObservation : +items </pre>			
Uses queries – Unit_patient_list_query, pat_session_URR_query			
Used by queries – pop_HD_session_info_query			

8.2.1.7 Patient Pre HD Urea Query

Query ID: {2A510CBA-7270-44ce-8895-0CA1C877C8C6}

Query Name: pre_HD_urea_query

Description - Return the pre blood urea of a patient given their unique system identifier. This query is run iteratively for each patient (or subject of care) id returned.

Rationale - The query is executed to return the pre haemodialysis blood urea value of a patient attending a particular renal dialysis unit if the instances of blood urea satisfy the given SNOMED CT (SCT) constraint expression and the observation value occurs within the specified time frame.

Query Type –

Reference Model Type: SubjectofCareEHR Query.

Function Type: Direct Query.

Query Parameter – The initial parameters are

Name	Value Type (ISO21090 data types)	Value (optional)	Description (optional)
Start.date.time	TS.DateTime	201003170900	The start date and time of the haemodialysis session.
Ehr.id	II		The shareable identifier of the patient who has had haemodialysis in a particular renal dialysis unit.
sct_blood_urea_constr	ST.CONSTR. SEMANTIC		The SCT constraint that defines the semantics of the blood urea instance to be queried

Query Range – The EHR which belongs to the patient attending a haemodialysis session queried iteratively.

Query Expression –

FOR \$z in (ehr.id.set)

FOR \$y in SubjectofCareEHR[@id=\$z]

FOR \$x in \$y/HaemodialysisSession/UreaReductionRatio/BloodUreaObservation

(: The parts of the EHR extract that needs to be queried :)

WHERE \$x/@meaning = (| 250623007 | blood urea measurement |) AND ((\$x/@obs_time.value > \$y/HaemodialysisSession/@session_time.low) AND (\$x/@obs_time.value < \$y/HaemodialysisTreatment/@obs_time.low))

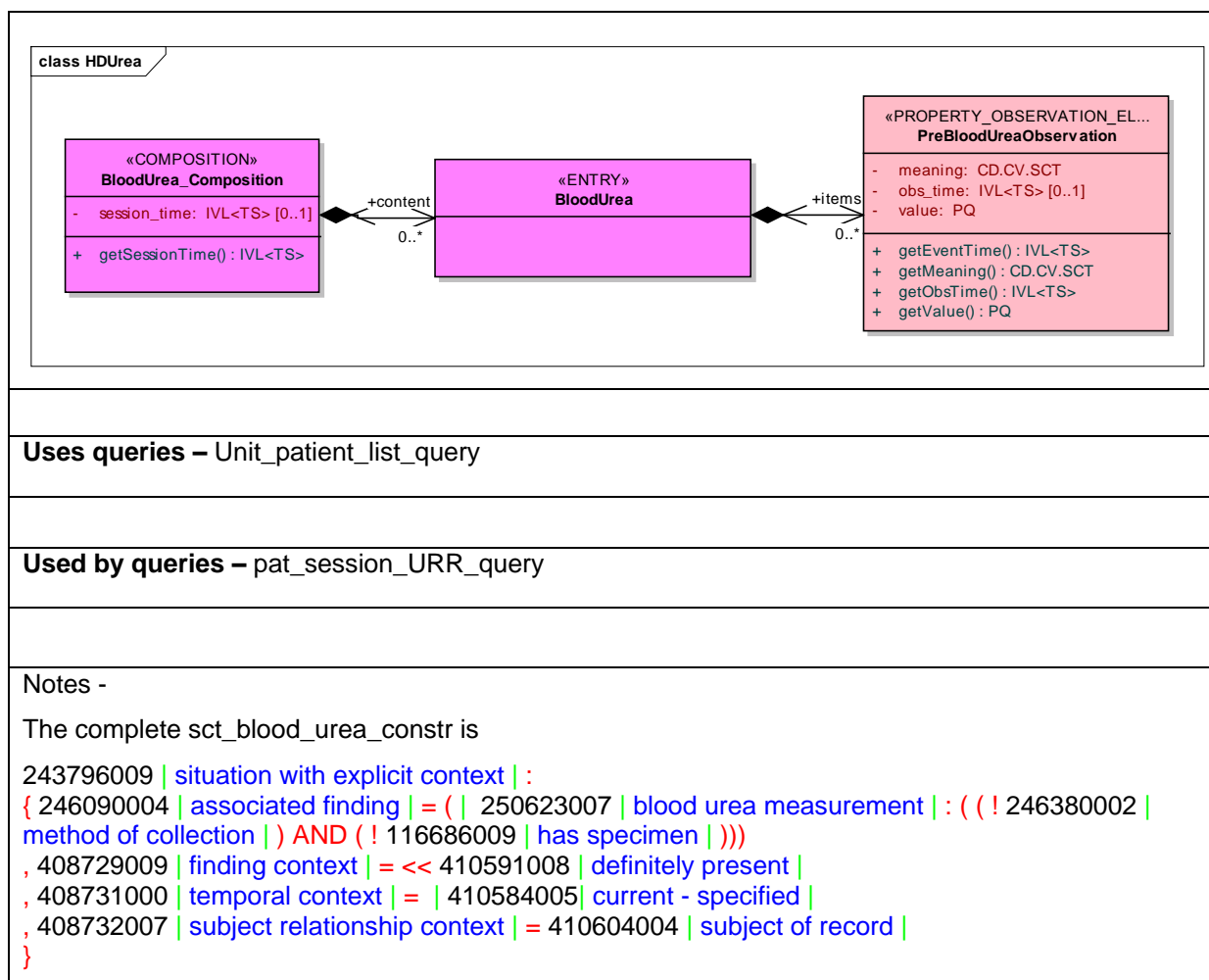
(: Check the patient record for elements with the blood urea that occur within the times stated in the predicate and satisfy the meaning stated in the SCT constraint expression. The complete constraint expression is available in the Notes section at the end of this query definition :)

LET \$a = \$x/value (: Store the pre blood urea value in an interim variable :)

RESULT

SubjectofCareEHR/URR_Composition/UreaReductionRatio/BloodUreaObservation/@value=\$a

(: Create a new composition with the result of the patient's pre blood urea :)



8.2.1.8 Patient Post HD Urea Query

Query ID: {2A510CBA-7270-44ce-8895-0CA1C877C8C6}

Query Name: post_HD_urea_query

Description - Return the post blood urea of a patient given their unique system identifier. This query is run iteratively for each patient (or subject of care) id returned.

Rationale - The query is executed to return the post haemodialysis blood urea value of a patient attending a particular renal dialysis unit if the instances of blood urea satisfy the given SNOMED CT (SCT) constraint expression and the observation value occurs within the specified time frame.

Query Type –

Reference Model Type: SubjectofCareEHR Query.

Function Type: Direct Query.

Query Parameter – The initial parameters are

Name	Value Type (ISO21090 data types)	Value (optional)	Description (optional)
End.date.time	TS.DateTime	201003171800	The end date and time of the haemodialysis session.
Ehr.id	II		The shareable identifier of the patient who has had haemodialysis in a particular renal dialysis unit.
sct_blood_urea_constr	ST.CONSTR. SEMANTIC		The SCT constraint that defines the semantics of the blood urea instance to be queried

Query Range – The EHR which belongs to the patient attending a haemodialysis session queried iteratively.

Query Expression –

FOR \$z in (ehr.id.set)

FOR \$y in SubjectofCareEHR[@id=\$z]

FOR \$x in \$y/HaemodialysisSession/UreaReductionRatio/BloodUreaObservation

(: The parts of the EHR extract that needs to be queried :)

WHERE \$x/@meaning = (| 250623007 | blood urea measurement |) AND ((\$x/@obs_time.value < \$y/HaemodialysisSession/@session_time.high) AND (\$x/@obs_time.value > \$y/HaemodialysisTreatment/@obs_time.high))

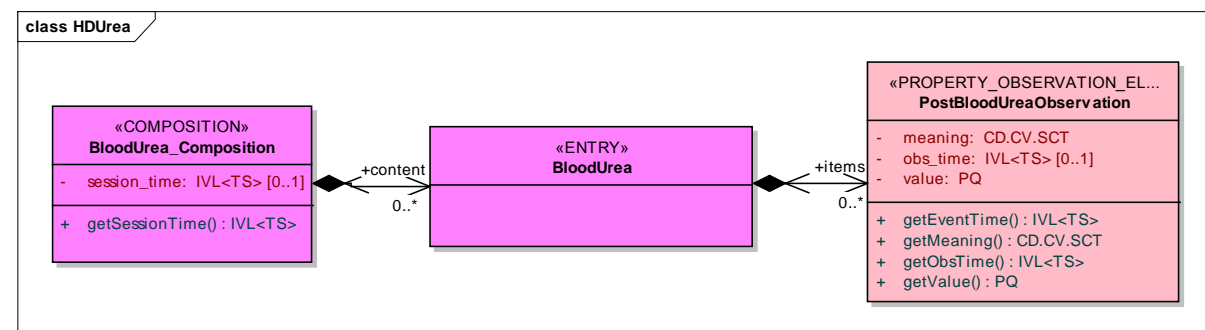
(: Check the patient record for elements with the blood urea that occur within the times stated in the predicate and satisfy the meaning stated in the SCT constraint expression. The complete constraint expression is available in the Notes section at the end of this query definition :)

LET \$a = \$x/value (: Store the pre blood urea value in an interim variable :)

RESULT

SubjectofCareEHR/URR_Composition/UreaReductionRatio/BloodUreaObservation/@value=\$a

(: Create a new composition with the result of the patient's post blood urea :)



Uses queries – Unit_patient_list_query

Used by queries – pat_session_URR_query**Notes -**

The complete sct_blood_urea_constr is

```
243796009 | situation with explicit context | :
{ 246090004 | associated finding | = ( | 250623007 | blood urea measurement | : ( ( ! 246380002 |
method of collection | ) AND ( ! 116686009 | has specimen | )))
, 408729009 | finding context | = << 410591008 | definitely present |
, 408731000 | temporal context | = | 410584005 | current - specified |
, 408732007 | subject relationship context | = 410604004 | subject of record |
}
```

8.2.2 Derived Operation Queries**8.2.2.1 Person BMI Query**

Query ID: {540E83D9-6F6A-470c-A367-9DF5827EDDB6}

Query Name: bmi_result_query

Description - Return the BMI of the subject of care given his/her weight and height.

Rationale – The query requires the calculation of a child's BMI which may or may not be persisted in his/her EHR.

Query Type –

Reference Model Type: SubjectofCareEHR Query.

Function Type: Derived Operation Query.

Query Parameter – The initial parameters are

Name	Value Type (ISO21090 data types)	Value (optional)	Description (optional)
Ehr.id	II		The shareable system identifier of the person.
height_result	COMPOSITION		The value of the height of the child present in a COMPOSITION
weight_result	COMPOSITION		value of the weight of the child present in a COMPOSITION

Query Range – The COMPOSITIONs which hold the height and weight values of the child.

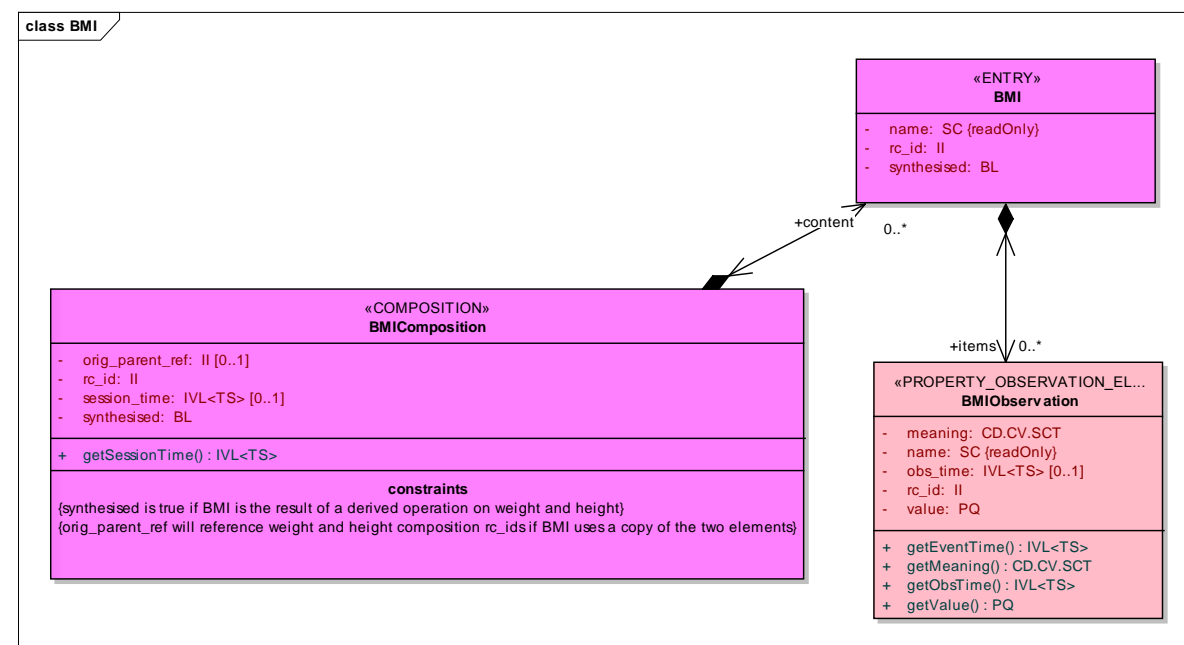
Query Expression –

FOR \$x in SubjectofCareEHR/COMPOSITION/BodyWeight_Entry/BodyWeightResult_Element
FOR \$y in SubjectofCareEHR/COMPOSITION/BodyHeight_Entry/BodyHeightResult_Element

LET \$a = \$x / \$y * \$y (: Equation to calculate BMI :)

RESULT SubjectofCareEHR/COMPOSITION/BMI_Entry/BMIObsResult_Element/@value=\$a

(: Create a new composition with the result of the person BMI :)



Uses queries – pct_Dob_Query, height_Query, weight_Query

Used by queries – population_obesity_query, obesity_finding_query

8.2.2.2 Patient Session URR

Query ID: {540E83D9-6F6A-470c-A367-9DF5827EDDB6}

Query Name: pat_session_URR_query

Description - Return the Urea Reduction Ratio of the patient given his/her pre and post blood urea for the session.

Rationale – The query requires the calculation of a patient's URR which may or may not then be persisted in his/her EHR.

Query Type –

Reference Model Type: SubjectofCareEHR Query.

Function Type: Derived Operation Query.

Query Parameter – The initial parameters are

Name	Value Type (ISO21090 data types)	Value (optional)	Description (optional)
Start.date.time	TS.DateTime	201003170900	The start date and time of the haemodialysis session.
End.date.time	TS.DateTime	201003171800	The haemodialysis session end date and time.
Ehr.id	II		The shareable identifier of the patient who has had haemodialysis in a particular renal dialysis unit.
PreHD_urea_result	COMPOSITION		The value of the pre HD urea of the patient present in a COMPOSITION
PostHD_urea_result	COMPOSITION		The value of the post HD urea of the patient present in a COMPOSITION

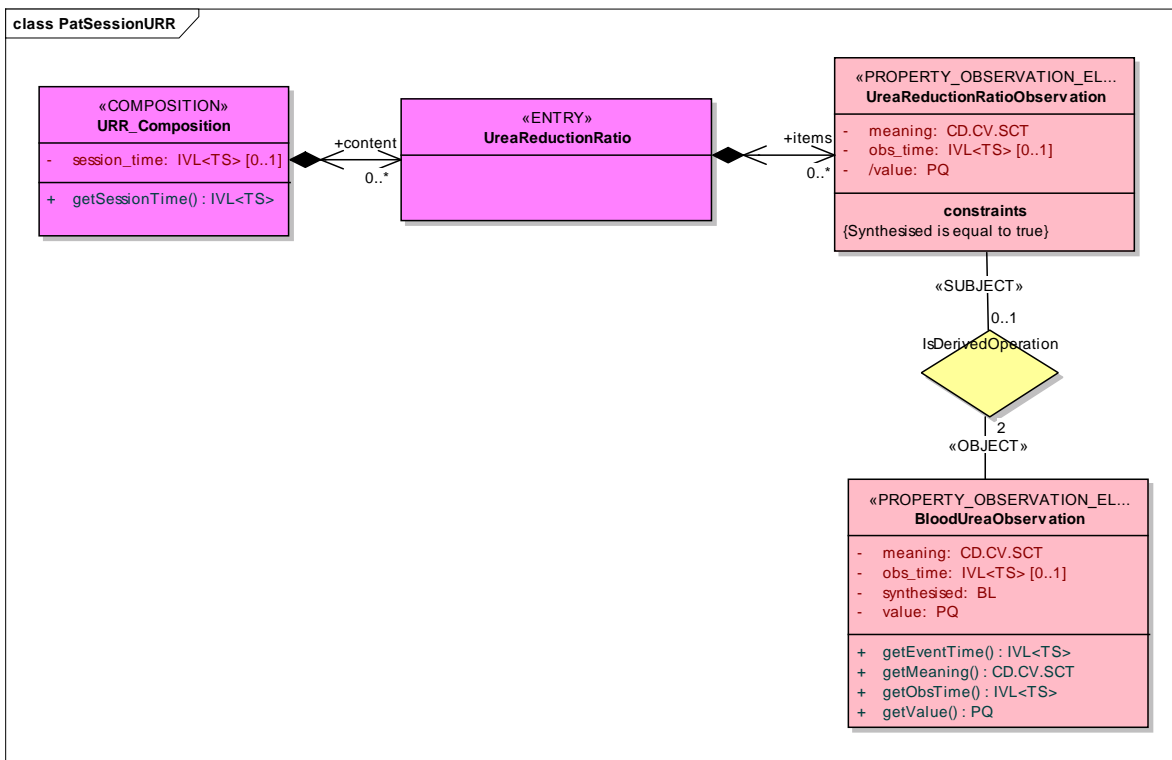
Query Range – The COMPOSITIONs which hold the pre and post HD urea values of the patient.**Query Expression –**

FOR \$x in SubjectofCareEHR/BloodUrea_Composition/BloodUrea/PreBloodUreaObservation
 FOR \$y in SubjectofCareEHR/BloodUrea_Composition/BloodUrea/PostBloodUreaObservation
 LET \$a = ((\$x - \$y) / \$x) * 100% (: Equation to calculate session URR :)

RESULT

SubjectofCareEHR/URR_Composition/UreaReductionRatio/UreaReductionRatioObservation/@value=\$a

(: Create a new composition with the result of the patient's session URR :)



Uses queries – Unit_patient_list_query, pre_HD_urea_query, post_HD_urea_query
Used by queries – patient_HD_session_info_query

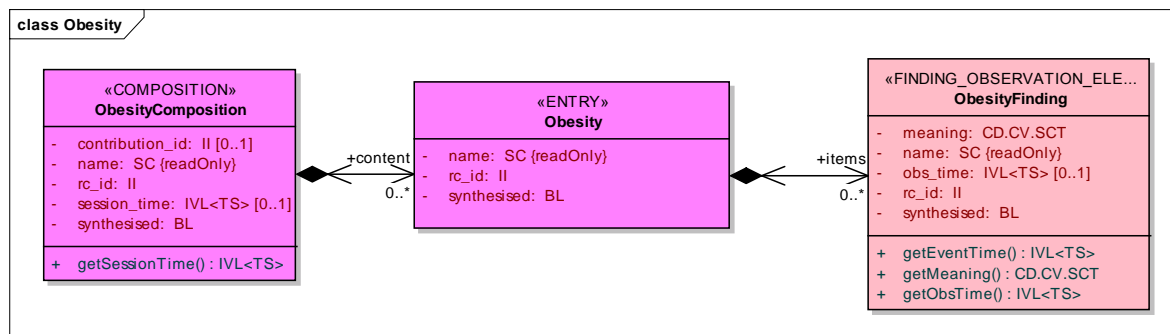
8.2.3 Direct Inference Queries

8.2.3.1 Person Obesity Finding Query

Query ID: {D0DEA5B8-8DDB-475b-9B4F-A55DA255D0A4}			
Query Name: obesity_finding_query			
Description – Determine whether a child has obesity given his/her BMI.			
Rationale - The National Child Measurement Programme requires summary details from Primary Care Trusts to monitor the number of reception year children who have been identified as obese.			
Query Type –			
Reference Model Type: SubjectofCareEHR Query.			
Function Type: Direct inference query.			
Query Parameter – The initial parameters are			
Name	Value Type (ISO21090 data types)	Value (optional)	Description (optional)
Ehr.id	II		The shareable system identifier of the person.
bmi_obs_result	COMPOSITION		The value of the calculated BMI of the person in the form of a COMPOSITION
Query Range – The COMPOSITION(s) which holds the bmi value of the child.			
Query Expression –			
FOR \$x in SubjectofCareEHR/COMPOSITION/BMI_Entry/BMIResult_Element			
RETURN if (\$x/@value >= 30)			
then			
SubjectofCareEHR/COMPOSITION/Obesity_Entry/ObesityFinding_Element/@value=true and			
SubjectofCareEHR/COMPOSITION/Obesity_Entry/ObesityFinding_Element/@meaning=(16286400			
5 body mass index 30+ - obesity (finding))			
else			
SubjectofCareEHR/COMPOSITION/Obesity_Entry/ObesityFinding_Element/@value=false			
SubjectofCareEHR/COMPOSITION/Obesity_Entry/ObesityFinding_Element/@meaning=(16286400			

5| [body mass index 30+ - obesity \(finding\)](#)|)

(: Create a new composition to indicate if the person is/isn't obese. For the complete SCT constraint expressions for the obesity finding refer to the Notes section at the end of this query definition. :)



Uses queries – pct_Dob_Query, bmi_result_query

Used by query – population_obesity_query

NOTES –

(a) The complete SCT constraint expression for an obesity finding present is

```

243796009 | situation with explicit context | :
{ 246090004 | associated finding | = 162864005 | body mass index 30+ - obesity (finding) |
, 408729009 | finding context | = 410515003 | known present |
, 408731000 | temporal context | = 410586007 | specified time |
, 408732007 | subject relationship context | = < 125676002 | person |
}
  
```

(b) The complete SCT constraint expression for an obesity finding absent is

```

243796009 | situation with explicit context | :
{ 246090004 | associated finding | = 162864005 | body mass index 30+ - obesity (finding) |
, 408729009 | finding context | = 410516002 | known absent |
, 408731000 | temporal context | = 410586007 | specified time |
, 408732007 | subject relationship context | = < 125676002 | person |
}
  
```

8.3 Population Queries

8.3.1 Direct Queries

8.3.1.1 Population Hypothyroidism Query

Query ID - {DED12DEE-3827-48d3-969D-19D9E41AF706}

Query Name - population_hypothyroidism_query

Description - Return the set of identified patients who have been diagnosed with hypothyroidism.

[Adapted from previous NHS Data Dictionary reporting request for obesity] [NCMP guidance for PCTs 2007-2008]

Rationale – The patient level pat_Med-based_diagnosis_query can be extended to include a cohort of patients to make it a population level query. This query returns a list of identified patients who have been diagnosed with hypothyroidism.

Query Type –

Reference Model Type: Population Query.

Function Type: Direct Query.

Query Parameter – The initial parameters are

Name	Value Type (ISO21090 data types)	Value (optional)	Description (optional)
Pat.id	SET<II>		Set of all unique identifiers of patients in a particular organisation or population group.
Hypothyroidism _diag_result	COMPOSITION		The COMPOSITION holding the information on whether a patient has been diagnosed with hypothyroidism.

Query Range – All EHRs which belong to a particular organisation.

Query Expression –

LET \$a := SET<II> (: Initialise output variable to store the list of unique patient identifiers with hypothyroidism :)

FOR \$z in (1 to n)<SET> (: Iterate through the input set of patient identifiers :)

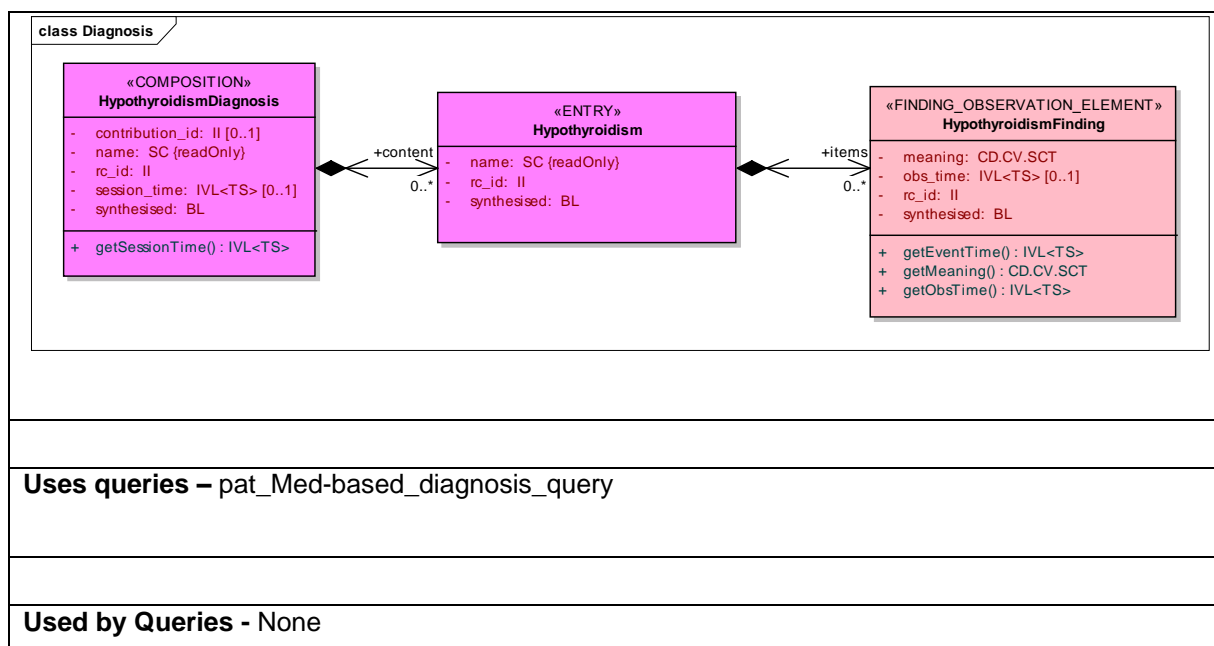
FOR \$x in SubjectofCareEHR/COMPOSITION/Diagnosis_Entry/HypothyroidismFinding_Element

FOR \$y in SubjectofCareEHR

RESULT if (\$x/@meaning = < 40930008 | hypothyroidism |)

then \$a = \$a.add(\$y/id)

(: If the patient is identified to have been diagnosed with hypothyroidism then add the patient system identifier to the set if not already present :)



8.3.1.2 Population HD Session Information Query

Query ID - {63F585B0-0E8E-4b1d-BFEF-F5B514FE9508}

Query Name – Pop_HD_session_info_query

Description - Return specific information regarding a haemodialysis session for all patients attending a renal dialysis unit.

[Knowledge Query extended to a Technical Query to demonstrate an end-to-end example]

Rationale – The query retrieves information on the pre and post diastolic and systolic blood pressure, as well as the dialyser surface area and URR for all patients attending a specific haemodialysis session in a renal dialysis unit. The session URR is calculated on the basis of the pre and post HD urea values.

Query Type –

Reference Model Type: Population Query.

Function Type: Direct Query.

Query Parameter – The initial parameters are

Name	Value Type (ISO21090 data types)	Value (optional)	Description (optional)
Start.date.time	TS.DateTime	201003170900	The start date and time of the haemodialysis session.
End.date.time	TS.DateTime	201003171800	The haemodialysis session end date and time.
Ehr.pat.ids	SET<II>		All the shareable identifiers of

			patients who have had haemodialysis in a particular renal dialysis unit.
Query Range – All EHRs which belong to the specific renal unit.			
<p>Query Expression –</p> <p>LET \$a : = SET<COMPOSITION> (: Initialise output variable to store all the COMPOSITIONs with haemodialysis session information of all patients attending a specific renal dialysis unit :)</p> <p>FOR \$z in (1 to n) <SET> (: Iterate through the input set of shareable patient identifiers ehr.pat.ids :)</p> <p>FOR \$x in SubjectofCareEHR/HaemodialysisSession[\$z] (: Collect the HD session COMPOSITION for patient \$z :)</p> <p>RESULT \$a = \$a.add(\$x)</p> <p>(: Add all the individual HD session COMPOSITIONs for each patient to a collection in order to present an aggregate result set for the entire population of patients :)</p>			
<p>class Haemodialysis Session</p>			
Uses queries – Unit_patient_list_query, patient_HD_session_info_query			
Used by Queries - None			

8.3.2 Derived Operation Queries

8.3.2.1 Population Obesity Query

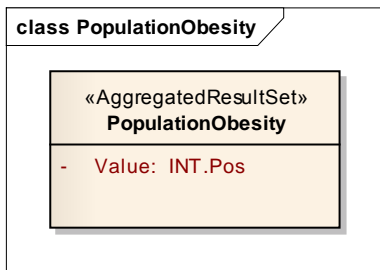
Query ID - {DED12DEE-3827-48d3-969D-19D9E41AF706}

Query Name - population_obesity_query

Description - Return the number of reception school children who have been identified as obese.

[Adapted from previous NHS Data Dictionary reporting request for obesity] [NCMP guidance for PCTs 2007-2008]

Rationale - The National Child Measurement Programme requires summary details from Primary Care Trusts to monitor the number of reception year children who have been identified as obese.

Information on the height, weight and age of all children measured is collated by the IC on behalf of the DH.			
Query Type – Reference Model Type: Population Query. Function Type: Derived Operation Query			
Query Parameter – The initial parameters are			
Name	Value Type (ISO21090 data types)	Value (optional)	Description (optional)
Org.code	II	ABC123	The unique identifier of the organisation to which a patient belongs or has some association. ABC123 is the code of the PCT responsible for schools in an area.
Obesity_finding_result	COMPOSITION		The COMPOSITION holding the information on whether a patient has been inferred to have obesity based on their BMI score.
Query Range – All EHRs which belong to the PCT with org.code=ABC123.			
Query Expression – LET \$a := INT_POS (0) FOR \$x in SubjectofCareEHR/COMPOSITION/Obesity_Entry/ObesityFinding_Element RESULT if (\$x/@value=true) then \$a = \$a + 1 (: Increment the counter for tracking population obesity by one each time a person is found to be obese :)			
 <pre> classDiagram class PopulationObesity { «AggregatedResultSet» - Value: INT.Pos } </pre>			
Uses queries – pct_Dob_Query, bmi_result_query, obesity_finding_query			
Used by Queries - None			

9 Recommendations

The query specification at present does not satisfy some of the query requirements stated in [2] shown below.

QUERY 1	Queries SHALL provide a simple method for testing existence and type of component relationships.
QUERY 2	Queries SHALL be able to test for equivalent component relationships types taking account of subsumption, inverses, symmetry and transitivity defined in the vocabulary.
QUERY 3	Queries SHALL provide a simple method for accessing linked components, including selecting them by type.

The query specification includes the provision for querying based on component relationship types. However, these component relationships have not yet been included in SNOMED CT to enable logic-based computations. It is proposed that upon inclusion of the relationship types in a logical hierarchy various methods can be applied to the LRA Reference Model record components based on subsumption, transitivity, inverse, and other description logic roles and relationships.

QUERY 4	A query development environment MIGHT be developed to support the authoring and testing of queries.
----------------	---

At present the query specification tests the LRA Query Model with the help of sequence diagrams (see Appendix A.1-A.3 for examples). However, a suitable development environment might be adopted or developed to enable the authoring and testing of LRA queries. Such a development tool should include the capability to query both SNOMED CT as well as the constrained LRA Reference Model also known as the LRA Interface Models. In addition, such a tool should also enable complex post-coordinated SNOMED CT queries in the form of SNOMED CT constraint expressions.

It is envisaged that the query specification will require modifications once the Demographics Model is formalised and necessary additions to SNOMED CT are finalised.

A Appendix

A.1 Patient Medication Query Illustration

The sequence diagram graphically presents the *medication-based patient diagnosis query* example. The diagram includes all the sub queries that are used to satisfy this SubjectofCareEHR query.

<<Diagram on the following page>>

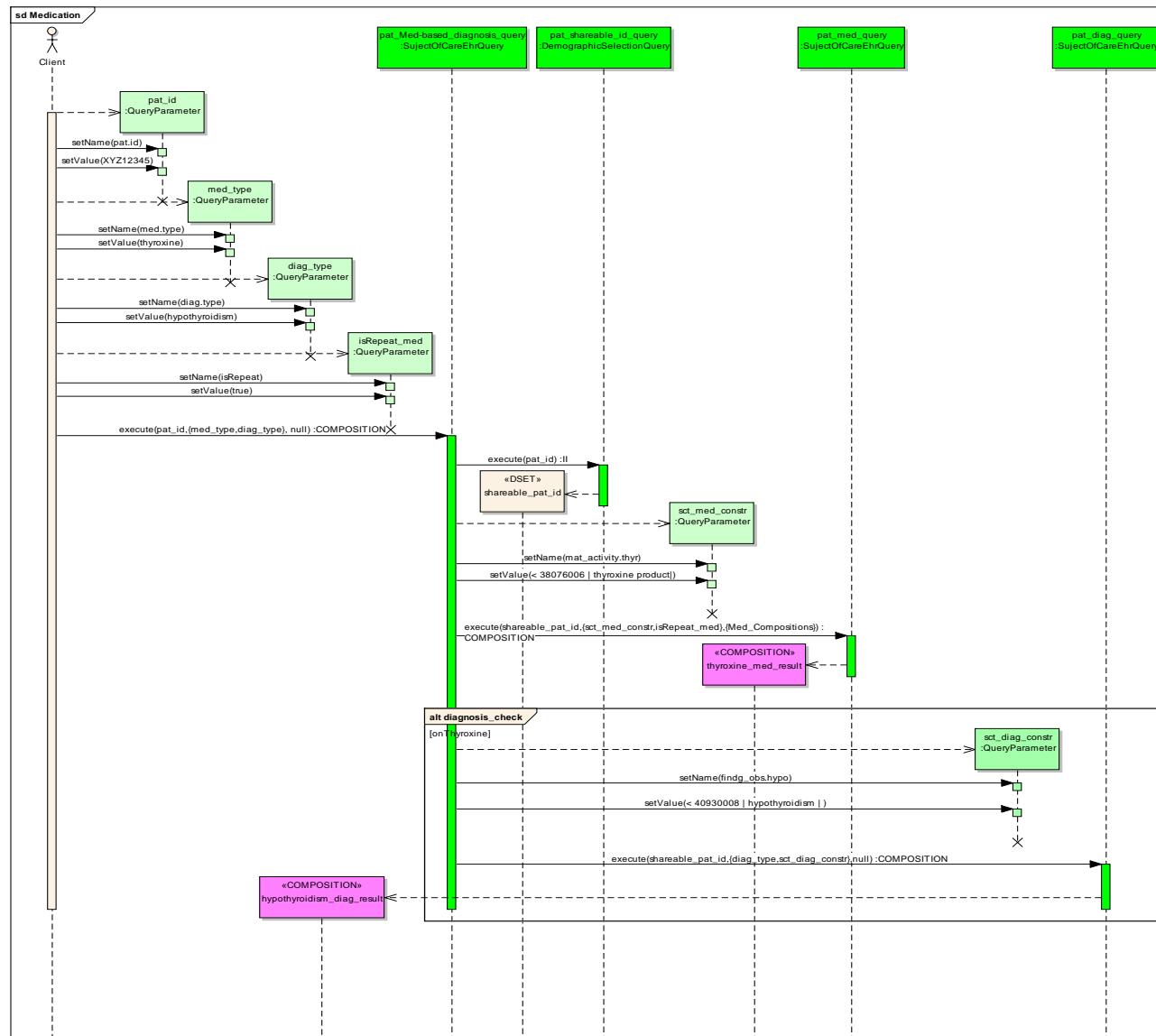


Figure A.1: Patient Medication-based Diagnosis example

A.2 Population Obesity Query Illustration

The sequence diagram graphically presents the *population obesity query* example. The diagram includes all the sub queries that are used to satisfy the population query.

<<Diagram on the following page>>

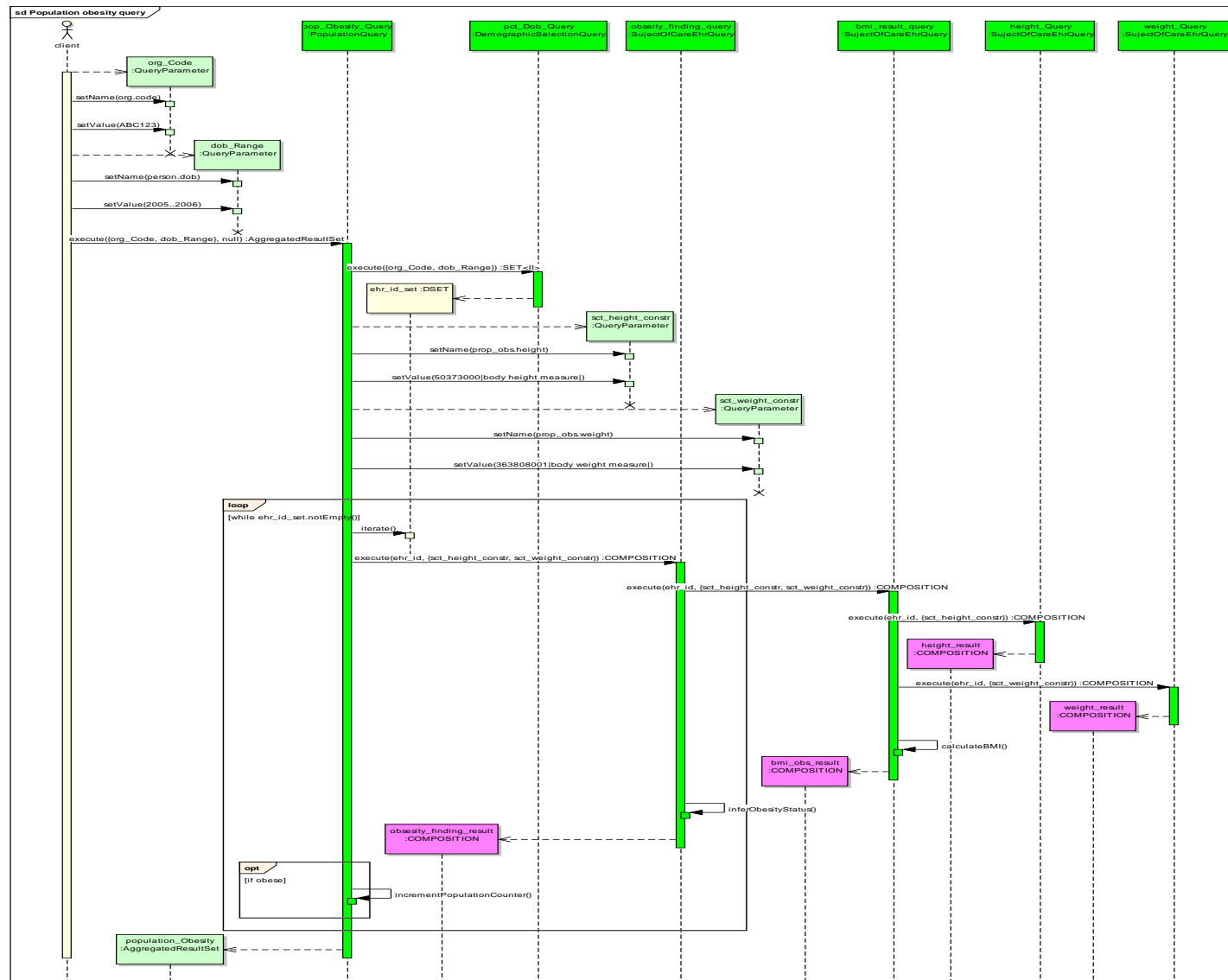


Figure A.2: Population Obesity Diagram

A.3 Population Haemodialysis Session Information Query Illustration

The sequence diagram graphically presents the *population haemodialysis session query* example. The diagram includes all the sub queries that are used to satisfy the population query.

<<Diagram on the following page>>

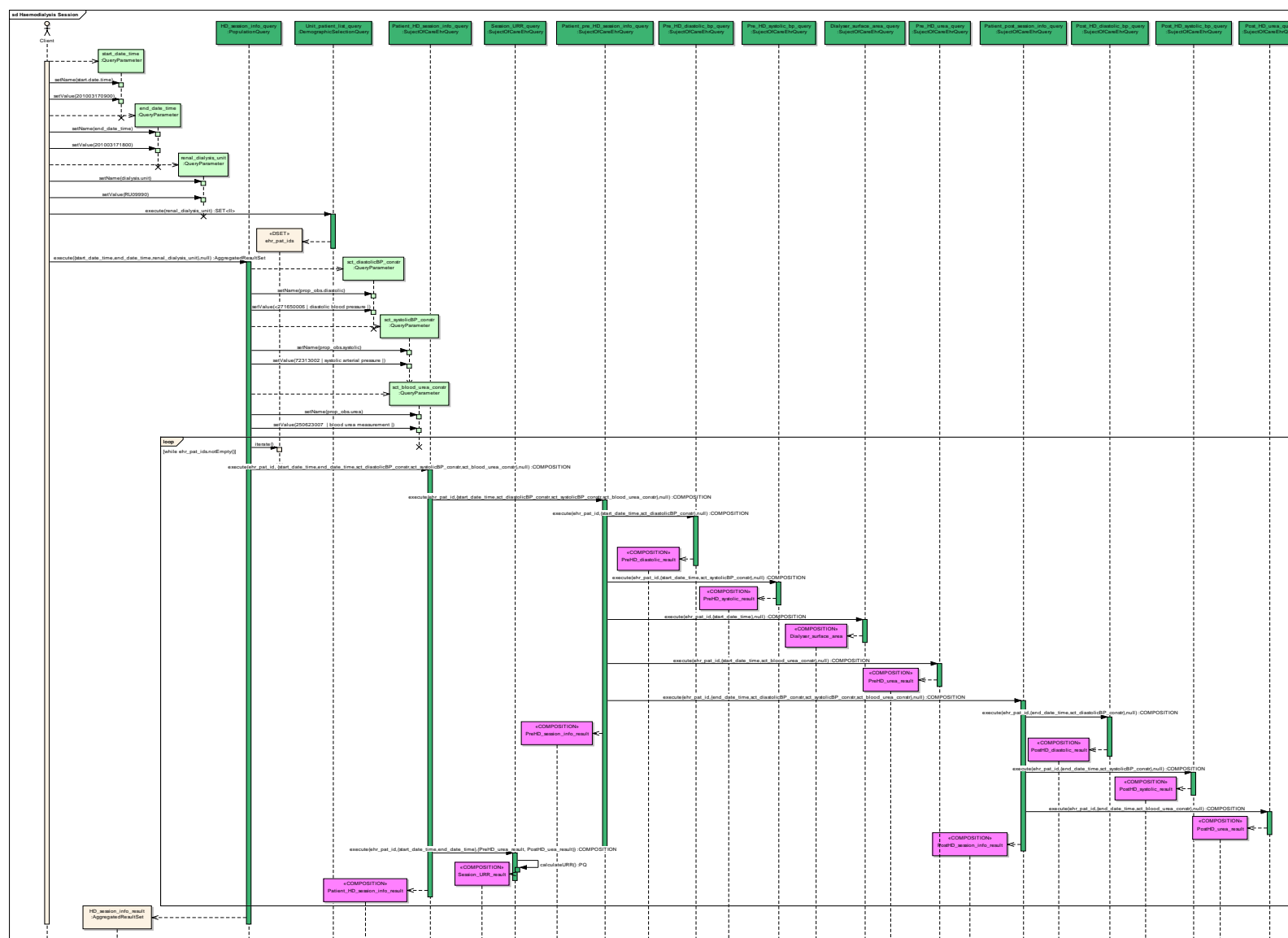


Figure A.3: Population Haemodialysis Session Information Diagram

A.4 LRA Technical Query Requirements Summary

QUERY 5	A Query MUST NOT allow the data of a care record to be modified.
QUERY 6	Queries MUST be uniquely identified.
QUERY 7	Queries SHOULD be able to be persisted in a searchable repository with appropriate access control restricting reading editing and authoring.
QUERY 8	Queries SHOULD have a name and/or short human-readable rendering that can be displayed in response to a search.
QUERY 9	Queries MUST have some human-readable format such that they can be reviewed against requirements to determine if they are suited to a particular use or reuse. Complex queries such as those made up of several sub-queries MAY have some level of drill-down so that more or less detail can be viewed.
QUERY 10	Query expressions SHOULD be able to include notes or comments inline clarifying the usage or operation of a particular section.
QUERY 11	The query repository SHOULD support versioning, change control and tracking of query authorship and modifications.
QUERY 12	Queries MUST allow specific terms to be marked as parameters which may be set at the point at which they are run.
QUERY 13	Queries MUST allow their input to be the output of another query.
QUERY 14	The output type of all queries MUST be clearly specified.
QUERY 15	A query development and assurance process MUST be available for use where query development is required.
QUERY 16	Query metadata MUST record the development status and assurance process stage of a query.
QUERY 17	Query metadata MUST record the point at which the query was reviewed with respect to the care record model. Where the model is subsequently changed this MUST be reviewed against dependant queries.
QUERY 18	Queries MUST be available in some representation accessible to clinicians for review.
QUERY 19	Query metadata MUST enumerate any assumptions made by the query author for the purpose of the query.
QUERY 20	Query metadata MUST separately record instances of query reuse.
QUERY 21	Where queries are reused, this reuse MUST be included within the query assurance process
QUERY 22	Queries MUST be able to be defined over a population or specific patient records.
QUERY 23	Where they have been used directly in the care of a particular patient, results of specific queries MUST be able to be stored within the care record of an individual patient alongside clinical entries.
QUERY 24	Collections retrieved from care records or returned from operations MUST be able to be sorted by one or more attributes.
QUERY 25	Collection sorting MUST be based on natural order and must apply to datatypes with a min operation defined.
QUERY 26	All other types MUST be partially ordered based on the equality semantics defined in the specification. Ordering of items with equal value is arbitrary, and MUST NOT be relied

	upon within queries.
QUERY 27	Queries MUST allow the sort order of nulls to be specified. When sorting ascending order and not otherwise specified, queries SHOULD default to sorting null flavors after non-null flavored data.
QUERY 28	Where query results represent a new clinical statement through calculating, generalising or inferring based on existing care record information, the query mechanism MUST be able to store this as a clinical statement in the care record, with links to the information on which the result is based.
QUERY 29	Where a query creates clinical statements, these statements MUST be marked as attested by the query system, along with other appropriate audit information.
QUERY 30	A query abstraction MUST define any aggregations and the logic of calculations and inferences based on record data selected by a query.
QUERY 31	Queries SHALL be able to return the entire care record for a patient matching a set of clinical and demographic characteristics
QUERY 32	Queries MUST be able to express literal SNOMED CT Expressions as query terms.
QUERY 33	Queries MUST implement the implies operation on CD as defined in the datatypes specification, such that where two SNOMED CT expressions have the same normal form they imply each other.
QUERY 34	Queries MUST be able to test if an instance of a SNOMED expression conforms to a SNOMED Expression Constraint.
QUERY 35	Queries MUST be able to represent SNOMED Expression Constraints as query terms.
QUERY 36	Queries SHOULD be able to accept all serializations of Expression Constraints as query terms.
QUERY 37	Queries SHOULD be able to construct SNOMED Expressions and Expression Constraints based on care record objects and query parameters.
QUERY 38	Queries SHALL be able to express terms using literal UCUM expressions
QUERY 39	Queries SHALL be able to return results in a specified unit where values are held or derived in a different, commensurate unit.
QUERY 40	Queries SHALL provide a simple method for testing existence and type of component relationships.
QUERY 41	Queries SHALL be able to test for equivalent component relationships types taking account of subsumption, inverses, symmetry and transitivity defined in the vocabulary.
QUERY 42	Queries SHALL provide a simple method for accessing linked components, including selecting them by type.
QUERY 43	A query MUST define its logic and output type but MUST NOT restrict implementation.
QUERY 44	A transformation from technical artefacts forming a query to an interface artefact including a UML activity diagram forming SHOULD be available.
QUERY 45	A query interface artefact MUST be available in order to represent the requirements specified in the knowledge space for a particular domain.
QUERY 46	A query MUST be able to specify the: <ul style="list-style-type: none"> identity or location of the record component attribute to which it applies; the logical comparator used to test the value of the attribute for inclusion in the result; and value of the predicate.
QUERY 47	Queries SHOULD allow definition of query sets based on results of other queries combined with basic set operations.

QUERY 48	A model abstraction representing the complete EHR of an individual patient or service user MUST be available against which to design and specify queries. This COULD be realised by an appropriately constrained instance of <code>Ira.technical.en13606.extract.EHR_EXTRACT</code> .
QUERY 49	The EHR model MUST use the same model for representing clinical statements and associated auditing, attestation and participant identification as the Care Components Model.
QUERY 50	Queries MUST, by default, be run against the latest version of an EHR.
QUERY 51	Queries MUST specify their result type.
QUERY 52	A query result SHALL contain either one or more <code>EHR_EXTRACT</code> instances each containing part or all of the EHR of an individual patient or service user or one or more aggregate result instances.
QUERY 53	Data type <code>CD.CV.SCT</code> MUST implement the logical comparator function <code>conformsTo(constraintRef: UUID, effectiveTime: TS): BL</code> which accepts a UUID that uniquely identifies the constraint expression constraint and an optional timestamp specifying the version.
QUERY 54	Data type <code>CD.CV.SCT</code> SHOULD implement the logical comparator function <code>conformsTo(constraintRef: II, effectiveTime: TS): BL</code> which accepts a SNOMED CT instance identifier that uniquely identifies the constraint expression constraint and an optional timestamp specifying the version.
QUERY 55	Data type <code>CD.CV.SCT</code> MUST implement the logical comparator function <code>conformsTo(constraint: ST): BL</code> which accepts an instance of a SNOMED CT constraint expression.
QUERY 56	Query implementations MUST handle dereferencing of expression constraints through retrieval from the constraint repository.
QUERY 57	Queries SHOULD use a common LRA metadata standard for storing, searching and accessing metadata on specific queries.
QUERY 58	Queries MUST be able to query using the enumerated values of record component attributes.
QUERY 59	Queries SHALL be able to test if an instance of ISO 21090-derived LRA datatype is a nullFlavor.
QUERY 60	Where appropriate, query operators MUST accept ISO21090 datatypes with null flavors, and follow the behaviour for returning a null flavour defined in the datatypes specification.
QUERY 61	Queries MUST provide the temporal comparators defined in ISO 12881 for the absolute temporal datatypes <code>TS</code> and <code>IVL(TS)</code> .
QUERY 62	Queries MUST be able to create an instance of <code>TS</code> containing the current point in time for use as a query term.
QUERY 63	Queries MUST be able to compare individual time and date components of a timestamp.
QUERY 64	The LRA query mechanism MUST implement the operators defined on the ISO 21090 datatypes used in the LRA.
QUERY 65	Query operations MUST accept ISO 21090 datatypes as parameters and MUST return an ISO 21090 datatype.
QUERY 66	Queries MUST NOT make use of operations on UML types.
QUERY 67	Queries MUST allow literal values of concrete datatypes to be used as query terms.
QUERY 68	Queries MUST provide a mechanism for creating instances of null flavors.

- | | |
|----------|--|
| QUERY 69 | Queries SHALL support the mathematical functions described in Error! Reference source not found.. |
| QUERY 70 | The query mechanism MAY define a method of extending the built in function set. |
| QUERY 71 | A query development environment MIGHT be developed to support the authoring and testing of queries. |