Informatics for Integrating Biology and the Bedside

i2b2 Process Document

Building Queries from Ontology Terms

Document Version: 1.6.1
i2b2 Software Version: 1.6

Table of Contents

DOCUMENT MANAGEMENT

Revision Number	Date	Author	Description of change
1.6.1	09/29/10	Lori Phillips	Initial document created.

1. BUILDING QUERIES FROM ONTOLOGY TERMS

In the i2b2 1.6 release we allow the direct use of the patient_dimension and the visit_dimension tables in a query built with the i2b2 query tool. The capability exists to some extent in earlier releases, but was not fully defined and implemented.

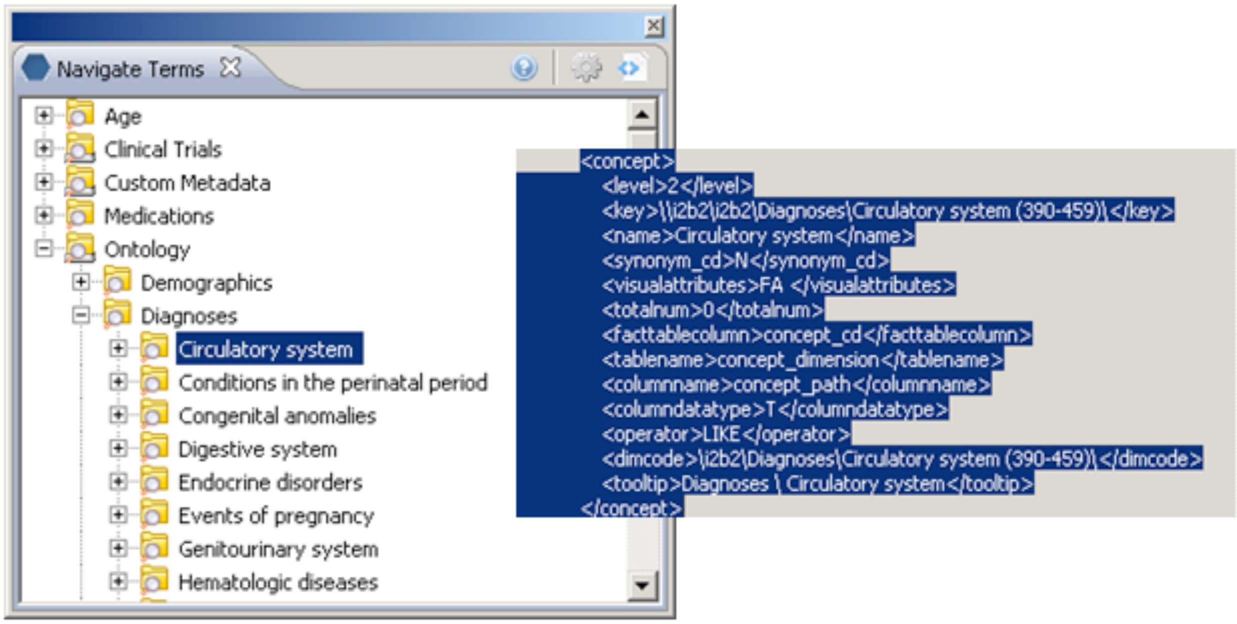
The ontology cell provides instructions on how the data repository cell should perform its query for any given term or set of terms. Each row in an ontology metadata table(s) represents a term or item, such as a diagnosis like “diabetes”, or a laboratory test like “white blood cell count”. In order to find these items in the data repository, the data repository cell needs to know which table to find them in and how to identify them. The information contained in the metadata tables is utilized to build a metadata SELECT SQL statement as follows:

```
select [c_facttablecolumnname] from [c_tablename]  where [c_columnname]
[c_operator] [c_dimcode]
```

The intent of this query is to associate a link between the dimension tables and the fact table for a given term. As a result every metadata SELECT SQL statement should return a fact table key.

A sample concept_dimension-based term is shown:

```
select concept_cd from concept_dimension where concept_path LIKE
'\i2b2\Diagnoses\Circulatory system (390-459)\%'
```



```
<concept>
   <level>2</level>
   <key>\\i2b2\i2b2\Diagnoses\Circulatory system (390-459)\</key>
   <name>Circulatory system</name>
   <synonym_cd>N</synonym_cd>
   <visualattributes>FA </visualattributes>
   <totalnum>0</totalnum>
   <facttablecolumn>concept_cd</facttablecolumn>
   <tablename>concept_dimension</tablename>
   <columnname>concept_path</columnname>
   <columndatatype>T</columndatatype>
   <operator>LIKE</operator>
   <dimcode>\i2b2\Diagnoses\Circulatory system (390-459)\</dimcode>
   <tooltip>Diagnoses \ Circulatory system</tooltip>
</concept>
```

The result of this query is a list of concept_cds to query against the fact table. The data repository cell uses this data to find a set of patients having a fact entry for this list of concept codes. This patient set finder query is as follows:

```
select patient_num from [c_facttablename] where [c_facttablecolumnname] IN
    (select [c_facttablecolumnname] from [c_tablename]  where [c_columnname]
    [c_operator] [c_dimcode])
```

Note that the sub-query in parentheses is the metadata SELECT SQL query.

Following through with our example, the patient set finder query would be:

```
select patient_num from observation_fact where concept_cd IN
    (select concept_cd from concept_dimension where concept_path LIKE
    '\i2b2\Diagnoses\Circulatory system (390-459)\%')
```

The ability to create a statement from the entries in the ontology metadata tables allows new items to be defined in a flexible manner. It also allows the ontology metadata table entries to have possible destructive effects if erroneous or malicious entries are added. Therefore, the trust relationship between the data repository cell and the ontology cell must be respected, and ontology metadata table entries must only be created by a trusted source.

2. EXTENDING QUERIES TO DIMENSION-BASED METADATA TERMS

2.1 Provider_dimension-based Metadata Terms

The ***Provider_dimension****-based metadata terms* are very similar in scope to concept_dimension-based terms: fact table key concept_cd is replaced by provider_id; metadata table's concept_path is replaced by provider_path. A sample patient set finder query is as follows:

```
select patient_num from observation_fact where provider_id in
    (select provider_id from provider_dimension where provider_path LIKE
    '\i2b2\Providers\Emergency\%')
```

2.2 Patient_dimension-based Metadata Terms

The ***Patient_dimension****-based metadata terms* are generally demographics-related: any of the following codes could be queried if they existed in the patient_dimension table:

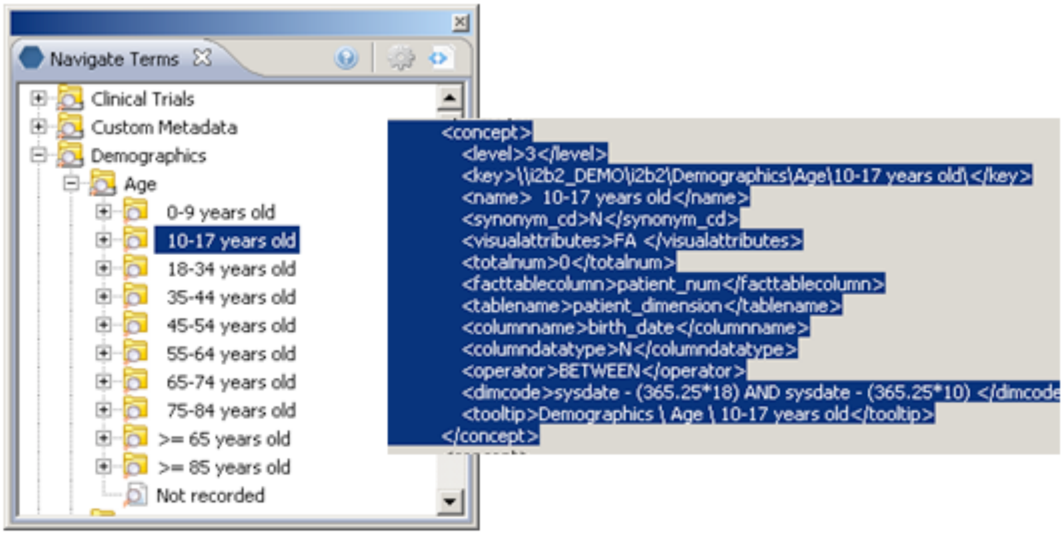
- sex_cd
- language_cd
- race_cd
- marital_status_cd
- religion_cd
- zip_cd
- statecityzip_path
- income_cd
- vital_status_cd

The associated metadata SELECT query is straightforward for each of these.

```
select patient_num from patient_dimension where race_cd = 'white'

select patient_num from patient_dimension where sex_cd = 'F'

select patient_num from patient_dimension where statecityzip_path LIKE
'Zip codes\Massachusetts\Cambridge\%'

select patient_num from patient_dimension where birth_date <  '01/01/2005'
```

Slightly more complicated are queries based on *birth_date* or *age*. The following identifies patients between the ages of 10 and 17 years old:

```
select patient_num from patient_dimension where birth_date BETWEEN
sysdate – (365.25*18)  AND sysdate – (365.25*10)
```



```
<concept>
   <level>3</level>
   <key>\\i2b2_DEMO\i2b2\Demographics\Age\10-17 years old\</key>
   <name>  10-17 years old</name>
   <synonym_cd>N</synonym_cd>
   <visualattributes>FA </visualattributes>
   <totalnum>0</totalnum>
   <facttablecolumn>patient_num</facttablecolumn>
   <tablename>patient_dimension</tablename>
   <columnname>birth_date</columnname>
   <columndatatype>N</columndatatype>
   <operator>BETWEEN</operator>
   <dimcode>sysdate - (365.25*18) AND sysdate - (365.25*10) </dimcode
   <tooltip>Demographics \ Age \ 10-17 years old</tooltip>
</concept>
```

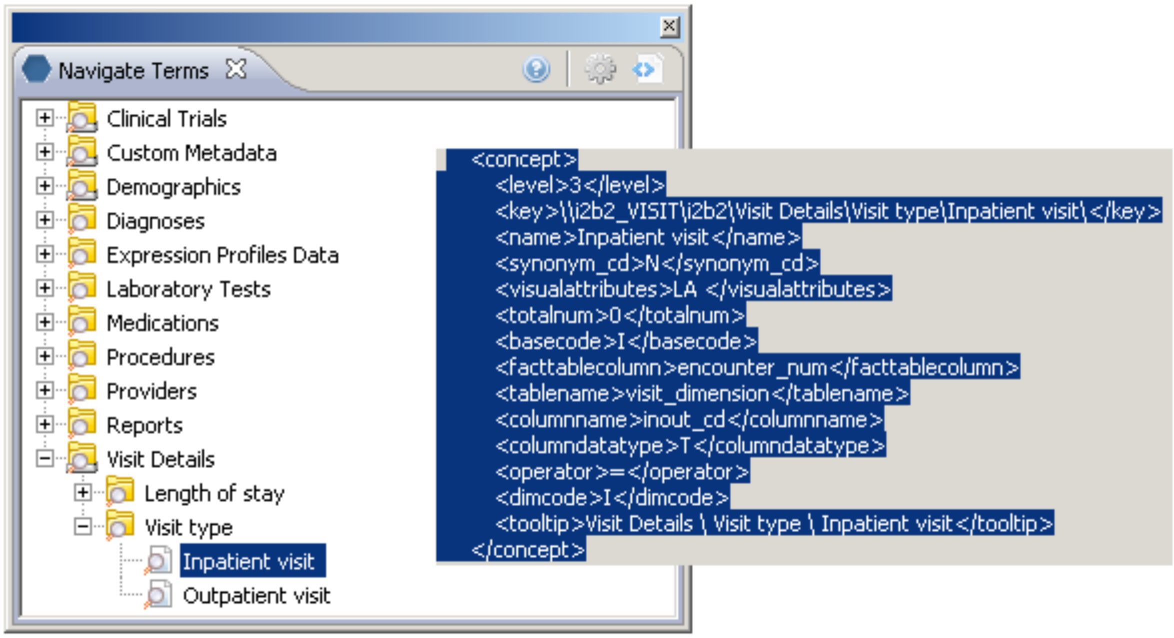
2.3 Visit_dimension-based Metadata Terms

The ***Visit_dimension****-based metadata terms* can be queried for any of the following codes if they exist in the visit_dimension table:

- active_status_cd
- inout_cd
- location_cd
- location_path
- length_of_stay

The associated metadata SELECT query is straightforward for each of these.

```
select encounter_num from visit_dimension where length_of_stay = 2

select encounter_num from visit_dimension where inout_cd = 'I'
```



```
 <concept>
    <level>3</level>
    <key>\\i2b2_VISIT\i2b2\Visit Details\Visit type\Inpatient visit\</key>
    <name>Inpatient visit</name>
    <synonym_cd>N</synonym_cd>
    <visualattributes>LA </visualattributes>
    <totalnum>0</totalnum>
    <basecode>I</basecode>
    <facttablecolumn>encounter_num</facttablecolumn>
    <tablename>visit_dimension</tablename>
    <columnname>inout_cd</columnname>
    <columndatatype>T</columndatatype>
    <operator>=</operator>
    <dimcode>I</dimcode>
    <tooltip>Visit Details \ Visit type \ Inpatient visit</tooltip>
 </concept>
```

3. DIMCODE RULES

Notice that the term’s dimcode passed by the ONT cell doesn’t necessarily match the [c_dimcode] used in the metadata SELECT SQL query 100%. In general, the dimcodes provided by the metadata table are allowed to lack punctuation required by the SQL statement if the “fixing” of the terms is unambiguous. The data repository cell remedies this according to a set of rules for each data type.

3.1 Text (T) Data Type

The following are the rules for the Text (T) data type:

# 1 For the **LIKE** operator, if the dimcode is missing ‘\’ at the end ***AND*** there are no enclosing single quotes ***AND*** there are no open and close parenthesis, ***then*** ‘\’ will be added at the end.

# 2 For the **LIKE** operator, if the dimcode is missing ‘%’ at the end ***AND*** there are no enclosing single quotes ***AND*** there are no open and close parenthesis, ***then*** ‘%’ will be added at the end.

# 3 For **any text** operator, enclosing single quotes will be added, if the dimcode is missing them ***AND*** there are no open and close parentheses.

# 4 For the **IN** operator, open and close parenthesis will be added, if the dimcode is missing them.

3.1.1 Text Data Examples

Operator	**Dimcode Format**	**Sample Metadata XML**	**SQL**

LIKE	Backslash, ‘%’ and enclosing single quotes will be added to the dimcode, if they are omitted	<concept>
		<facttablecolumn>patient_num</facttablecolumn>
		<tablename>patient_dimension</tablename>
		<columnname>statecityzip_path</columnname>
		<columndatatype>T</columndatatype>
		<operator>LIKE</operator>
		<dimcode>\MA\Charlestown</dimcode>
		..
		</concept>
			SELECT patient_num
			FROM patient_dimension
			WHERE statecityzip_path
			LIKE '\MA\Charlestown\%'

IN	(‘val1,’val2’,’val3’) or
	‘val1’,’val2’,’val3’
	Enclosing parentheses will be added to the dimcode if they are omitted
		<concept>
		<facttablecolumn>patient_num</facttablecolumn>
		<tablename>patient_dimension</tablename>
		<columnname>race_cd</columnname>
		<columndatatype>T</columndatatype>
		<operator>IN</operator>
		<dimcode>’NA’,’NAT. AM.’</dimcode>
		..
		</concept>
			SELECT patient_num
			FROM patient_dimension
			WHERE race_cd IN
			('NA','NAT.AM.')

All text Operators	‘val1’ or val1
	Enclosing single quotes will be added to the dimcode if they are omitted
		<concept>
		<facttablecolumn>patient_num</facttablecolumn>
		<tablename>patient_dimension</tablename>
		<columnname>race_cd</columnname>
		<columndatatype>T</columndatatype>
		<operator>=</operator>
		<dimcode>NAT. AM</dimcode>
		..
		</concept>
			SELECT patient_num
			FROM patient_dimension
			WHERE race_cd = 'NAT.AM'

3.2 Number (N) Data Type

The following are the rules for the Text (T) data type:

# 1 For the **IN** operator, open and close parenthesis will be added if the dimcode is missing them.

3.2.1 Numeric Data Examples

Operator	Dimcode Format	Sample Metadata XML	SQL

IN	(val1,val2,val3) **or** val1,val2,val3
	Enclosing parentheses will be added to the dimcode if they are omitted
		`<concept>`
		`<facttablecolumn>patient_num </facttablecolumn>`
		`<tablename>patient_dimension</tablename>`
		`<columnname>AGE_IN_VALUE_NUM</columnname>`
		`<columndatatype>N</columndatatype>`
		`<operator>IN</operator>`
		`<dimcode>0,5,10</dimcode>`
		..
		`</concept>`
			SELECT patient_num
			FROM patient_dimension
			WHERE age_in_value_num
			IN (0,5,10)

3.3 Date (D) Data Type

The following are the rules for the Date (D) data type:

# 1 For the **IN** operator, open and close parenthesis will be added if the dimcode is missing them.

3.3.1 Date Data Examples

Operator	Dimcode Format	Sample Metadata XML	SQL

IN	(‘val1’,’val2’,‘val3’) **or** ‘val1’,’val2’,‘val3’
	The open and close braces enclosing the list of date values are optonal.
		For SQLServer:
		<concept>
		<facttablecolumn>patient_num
		</facttablecolumn>
		<tablename>visit_dimension</tablename>
		<columnname>start_date</columnname>
		<columndatatype>D</columndatatype>
		<operator>IN</operator>
		<dimcode>**’2010-10-10 13:30:00’ , ’2009-09-10 13:30:00’**
		</dimcode>
		..
		</concept>
			SELECT patient_num
			FROM visit_dimension
			WHERE start_date
			IN ('2010-10-10 13:30:00', '2009-09-10 13:30:00')

All date operators (= , > , < , <> , BETWEEN)	Follows the database specific value format related to the date operator	**For SQLServer:**
		<concept>
		<facttablecolumn>patient_num
		</facttablecolumn>
		<tablename>patient_dimension</tablename>
		<columnname>birth_date</columnname>
		<columndatatype>D</columndatatype>
		<operator>=</operator>
		<dimcode>**’2010-10-10 13:30:00’**</dimcode>
		..
		</concept>
		For Oracle:
		<concept>
		<facttablecolumn>patient_num
		</facttablecolumn>
		<tablename>patient_dimension</tablename>
		<columnname>birth_date</columnname>
		<columndatatype>D</columndatatype>
		<operator>=</operator>
		<dimcode>**to_date('01-JAN-1998', 'DD-MON-YYYY')**
		</dimcode>
		</concept>
			For SQL Server:
			SELECT patient_num
			FROM patient_dimension
			WHERE birth_date = ('2010-10-10 13:30:00')
			For Oracle:
			SELECT patient_num
			FROM patient_dimension
			WHERE birth_date = to_date ('01-JAN-1998, DD-MON-YYYY')