



i2b2 Cell Messaging
Data Repository (CRC) Cell

Document Version: 1.5.2
I2b2 Software Release: 1.5

Table of Contents

1. Introduction	4
1.1 The i2b2 Hive	4
1.2 i2b2 Messaging Overview	4
1.2.1 XSD Files for i2b2 XML Schema	5
1.2.1.1 i2b2.xsd	5
1.2.1.2 i2b2_request.xsd	5
1.2.1.3 i2b2_response.xsd	5
2. Data Repository (CRC) Cell Messaging Detail	6
2.1 Use Cases	7
2.2 Services / Messages	8
2.2.1 Browse Query History	8
2.2.2 Run Patient Set Query	8
2.2.3 Get Patient Data Query	9
2.3 Patient Set Query Service	9
2.3.1 Conceptual Model:	9
2.3.2 Panel Definition Details	10
2.3.2.1 Request and Response message structure	13
2.3.3 Request and Response Object Model	14
2.3.4 Use Case Scenarios	16
2.3.4.1 Execute a query and get its results	16
2.3.4.2 Message Request and Response	19
2.3.4.3 Scenario: Check if the query is completed and get its results	22
2.3.4.4 Scenario: Get the query result (xml output) by result instance	23
2.3.4.5 Scenario: Run an Analysis Plug-in and get the status	25
2.3.4.6 Scenario: Get a list of queries by user id	27
2.3.4.7 Scenario: Get query definition from master id	29
2.3.4.8 Scenario: Rename a query	30
2.3.4.9 Scenario: Delete a query	31
2.3.4.10 Scenario: Cancel a query	32
2.4 Patient Data Object Query Service	34
2.4.1 Request and Response message structure	34
2.4.2 Request and Response Object Model	35
2.4.3 Use Case Scenario	36
2.4.3.1 Scenario: Get patient data from a patient set id	36
2.4.3.2 Scenario: Get Observation blob by primary key	46
2.5 Data Upload Messages	47
2.5.1 i2b2 Import: Append Flag	48
2.5.2 Use Case Scenarios	49
2.5.2.1 Scenario: Upload data in Patient Data Object XML	49
2.5.2.2 Scenario: Get Upload status info by upload_id and user_id	55
2.6 Message Explanations	55
2.6.1 Header	56
2.6.2 Requests	56
2.6.2.1 xsi:type="crc:user_requestType"	56
2.6.2.2 xsi:type="crc:master_requestType"	57

2.6.2.3	xsi:type="crc:instance_requestType" _____	57
2.6.2.4	xsi:type="crc:query_definition_requestType" _____	57
2.6.2.5	xsi:type="crc:getPDO_fromInputList " _____	58
2.6.2.6	xsi:type="crc: GetObservationFactByPrimaryKey_requestType " _____	59
2.6.3	Responses _____	59
2.6.3.1	xsi:type="crc:master_responseType" _____	59
2.6.3.2	xsi:type="crc:instance_responseType" _____	60
2.6.3.3	xsi:type="crc:query_result_instanceType" _____	61
2.6.3.4	xsi:type="crc:request_xml_responseType" _____	62
2.6.3.5	xsi:type="crc:master_instance_result_responseType" _____	62
2.6.3.6	xsi:type="crc:instance_result_responseType" _____	64
2.6.3.7	xsi:type="crc:patient_data_responseType" _____	65
2.7	XML Schema Definitions _____	65
2.7.1	CRC.xsd _____	66
2.7.2	CRC_PDO_QRY.xsd _____	66
2.7.3	CRC_PDO_QRY_request.xsd _____	66
2.7.4	CRC_PDO_QRY_response.xsd _____	66
2.7.5	CRC_PSM_OBJ.xsd _____	66
2.7.6	CRC_PSM_QRY.xsd _____	66
2.7.7	CRC_PSM_QRY_request.xsd _____	66
2.7.8	CRC_PSM_QRY_response.xsd _____	66
2.7.9	CRC_PSM_QRY_query_definition.xsd _____	66
2.7.10	CRC_PANEL.xsd _____	66
2.7.11	CRC_UPLOADER_QRY.xsd _____	67
2.7.12	I2B2_RESULT_MSG.xsd _____	67

1. INTRODUCTION

This document gives an overview of i2b2 cell messaging as well as a more detailed description of message formats specific to the Data Repository (CRC) Cell.

1.1 The i2b2 Hive

The Informatics for Integrating Biology and the Bedside (i2b2) is one of the sponsored initiatives of the NIH Roadmap National Centers for Biomedical Computing (<http://www.bisti.nih.gov/ncbc/>). One of the goals of i2b2 is to provide clinical investigators broadly with the software tools necessary to collect and manage project-related clinical research data in the genomics age as a cohesive entity – a software suite to construct and manage the modern clinical research chart. The i2b2 hive is a set of cells or modules that have a common messaging protocol that allow the cells to interact using web services and XML messages.

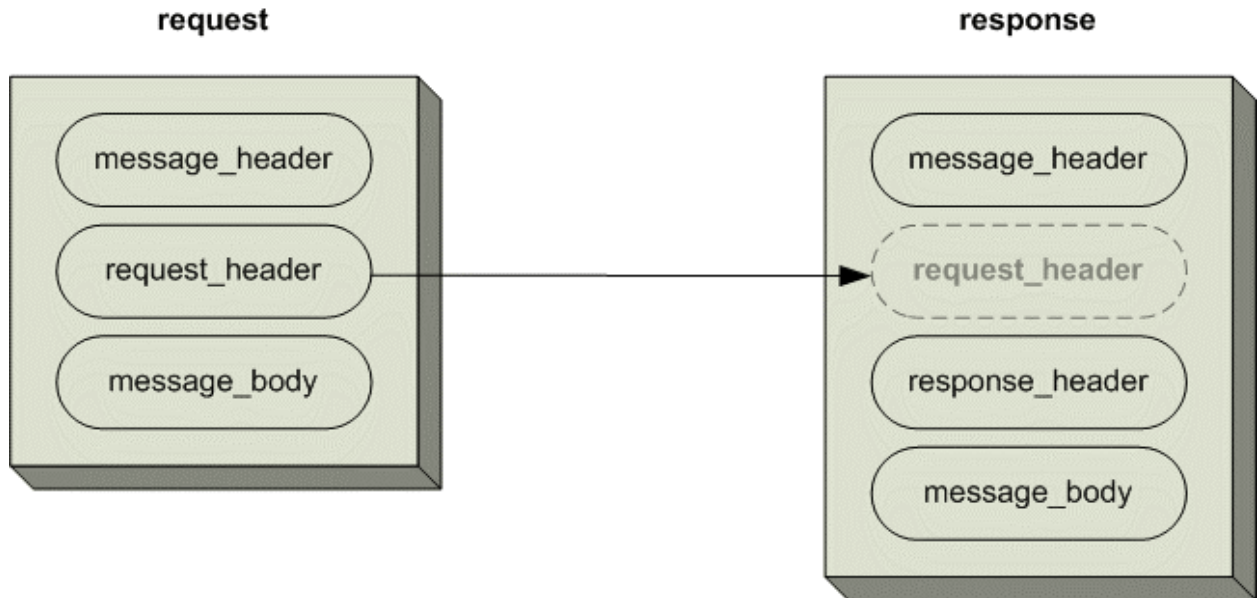
1.2 i2b2 Messaging Overview

All cells in the i2b2 hive must communicate using standard i2b2 XML messages. This message specifies certain properties that are common to cells and essential to the administration tasks associated with sending, receiving and processing messages.

All requests are sent using a `<request>` tag and responses are returned using a `<response>` tag. The same `<message_header>` tag is used for both. The `<request_header>` is used for requests but may optionally be echoed back in the response. The response must include a `<response_header>`.

The XSD specification of the i2b2 message permits individual cells to add cell-specific XML in the `<message_body>` tag. This cell-specific XML need not extend the i2b2 message schema since the i2b2 schema will allow insertion of tags from any namespace into the `<message_body>` tag.

The following image illustrates the basic top-level elements contained within the request and response messages.



1.2.1 XSD Files for i2b2 XML Schema

The i2b2 XML schema consists of three XSD files:

1.2.1.1 I2B2.XSD

This schema is not used directly to create i2b2 messages, but is included in the i2b2_request.xsd and the i2b2_response.xsd. It defines the <message_header> tag.

1.2.1.2 I2B2_REQUEST.XSD

This schema is used for validating i2b2 request messages. It defines the <i2b2:request> tag, which includes the <message_header> tag.

1.2.1.3 I2B2_RESPONSE.XSD

This schema is used for validating i2b2 request messages. It defines the <i2b2:response> tag, which includes the <message_header> tag.

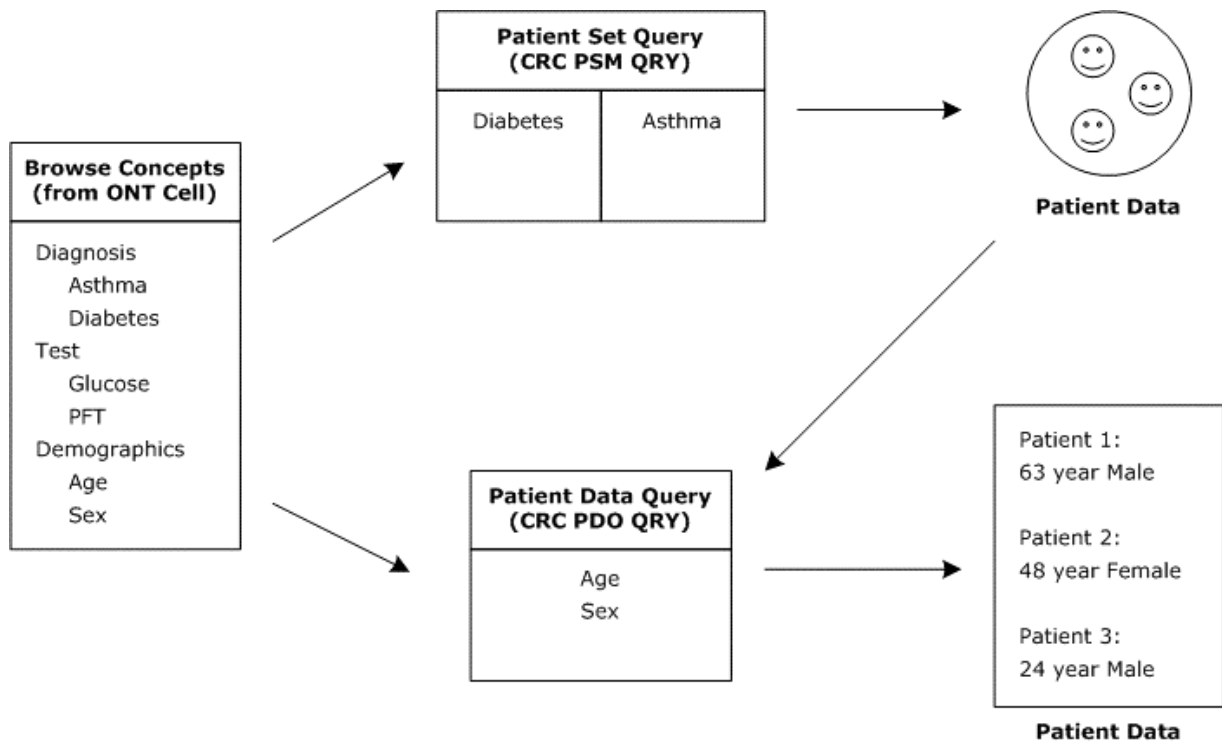
2. DATA REPOSITORY (CRC) CELL MESSAGING DETAIL

The Data Repository Cell is one of the core cells in the i2b2 hive. Since much of the data in the repository is clinical in nature, it has also come to be known as the Clinical Research Chart (CRC) and the terms “**data repository**” and “**CRC**” are used interchangeably. The data repository is a warehouse of patient phenotypic and genotypic data that interacts with other cells to provide information for users.

Communication with the CRC Cell, like all cells in the i2b2 hive, is handled via standardized XML web services. These XML messages conform to the i2b2 messaging standard described above, which allows cell-specific XML within the <message_body> tag. The remainder of this document describes CRC-specific web services and the XML formats that encode them and illustrates how these XML messages are used to accomplish a set of interactions that correspond to typical CRC use cases.

A typical CRC client may want to define a patient set and then request patient data on that set. Both of these tasks require the user to first interact with another cell called the Ontology Management (ONT) Cell in order to choose concept codes to define the CRC request. Although the specific interactions with the ONT Cell are not described in this document, the following diagram shows the basic flow of information.

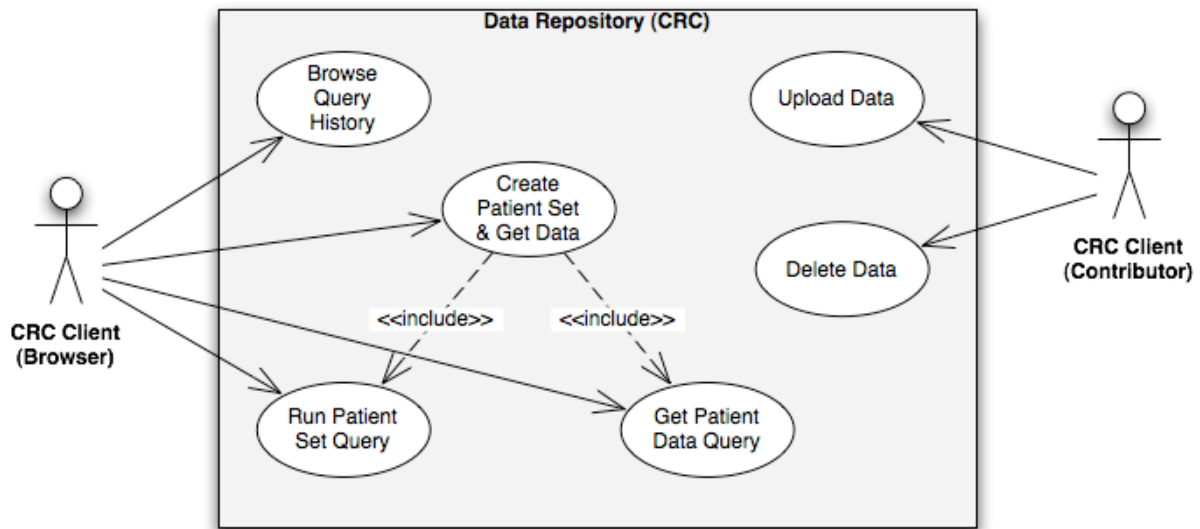
This diagram describes how a user may get the ages and genders for all patients who have either diabetes or asthma. The user starts by selecting the diagnoses ‘Diabetes’ and ‘Asthma’ from the ONT cell; these define the ‘**Patient Set Query**’, which creates a **patient set** in the data repository. Then the user selects the demographic concepts ‘Age’ and ‘Sex’ from the ONT cell to define the ‘**Patient Data Query**’. The patient data query returns the age and gender for all patients in the data set, those with diabetes or asthma.



2.1 Use Cases

The CRC Cell is a repository of clinical data and has a set of services that respond to requests for patient data. A request might be issued by a client cell which is used by a researcher conducting a clinical trial in order to help gather a cohort.

There are two types of clients or users, the '**browsing client**' and the '**contributing client**'. The contributing client adds content to the CRC by uploading patient data or deleting data by removing previous uploads. The browsing client has four possible interactions with the repository cell. The user may create queries that define patient sets, browse previous queries, rerun existing patient set queries and get specific patient data from a patient set.



2.2 Services / Messages

The CRC Cell provides services that support the interactions necessary for each of the use cases described in the previous section. The services expect different message request types for each specific behavior or request.

2.2.1 Browse Query History

Get a List of Saved Query Definitions	provides a list of all prior queries created by a client/user
Get a List of Saved Query Results	provides query results for given query run/instance
Get an XML Definition of a Defined Query	returns definition of the query

2.2.2 Run Patient Set Query

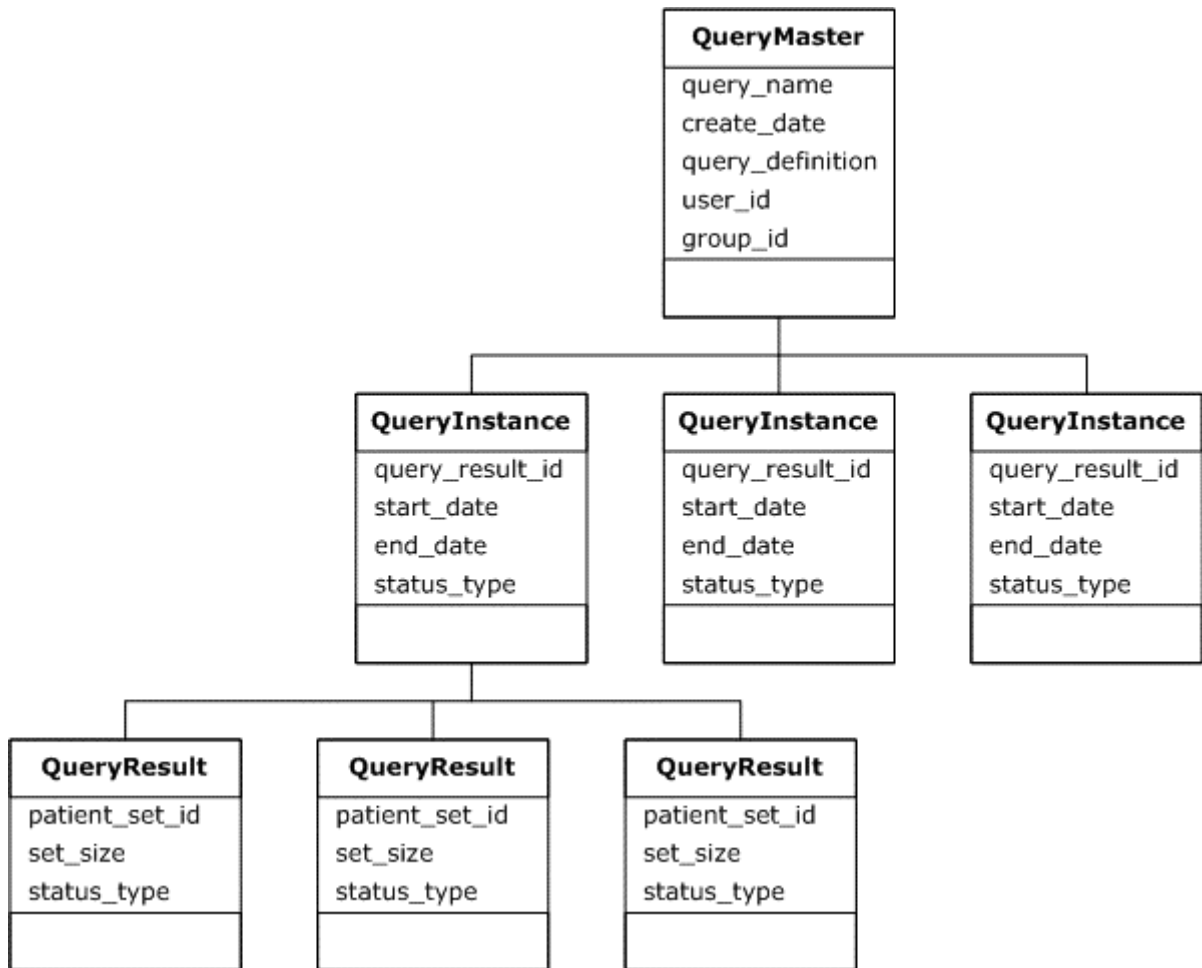
Run Patient Set Query	runs the query and returns the result and its status
-----------------------	--

2.2.3 Get Patient Data Query

Get Patient Data From a Patient Set	returns patient data object for the given patient set
-------------------------------------	---

2.3 Patient Set Query Service

2.3.1 Conceptual Model:



The **QueryMaster** holds master information about the query like the query name, query_definition, user_id, and group_id. The **query_definition** element in QueryMaster holds the xml representation of the query constraint; its details are described below.

The run information of the query is recorded in QueryInstance and is created whenever the query is executed. There can be multiple QueryInstances for one QueryMaster and one QueryInstance can have multiple QueryResults. QueryResult typically holds result information of the query, such as the id and the set size of the patient set, as well as the result status.

2.3.2 Panel Definition Details

The **panel** is the common definition used to query the data mart. The panel holds one or more <items>. The <item> constraints are OR-ed and the <panel> constraints are AND-ed in the query. Both the setfinder and PDO request share this panel definition.

Example:

Italicized and *Gray* are optional for both setfinder and PDO schema.

Gray items (no italics) are required only if the parent container is being used.

Italicized items are optional but **only** for setfinder schema.

All others are required.

```
<panel name="name0">
  <panel_number># of panel</panel_number>
  <panel_date_from time="start_date" inclusive="yes"></panel_date_from>
  <panel_date_to time="start_date" inclusive="yes"></panel_date_to>
  <panel_accuracy_scale>1</panel_accuracy_scale>
  <invert>0</invert>
  <total_item_occurrences operator="EQ/NE/GT/GE/LT/LE">
1</total_item_occurrences>
  <item>
    <hlevel>3</hlevel>
    <item_name>item_name0</item_name>
    <item_table>
      <item_key>item_key0</item_key>
      <item_icon>item_icon0</item_icon>
      <tooltip>tooltip0</tooltip>
      <class>ENC</class>
      <constrain_by_value>
        <value_operator>EQ/NE/GT/GE/LT/LE/IN/BETWEEN/LIKE</value_operator>
```

```

        <value_constraint>value_constraint0</value_constraint>
        <value_unit_of_measure>unit</value_unit_of_measure>
        <value_type>TEXT/NUMBER/FLAG/MODIFIER</value_type>
    </constrain_by_value>
    <constrain_by_date>
        <date_from time="start_date" inclusive="yes">2006-05-04</date_from>
        <date_to time="start_date" inclusive="yes">2006-05-04</date_to>
    </constrain_by_date>
    <dim_tablename>dim_tablename0</dim_tablename>
    <dim_columnname>dim_columnname0</dim_columnname>
    <dim_dimcode>dim_dimcode0</dim_dimcode>
    <dim_columndatatype>dim_columndatatype0</dim_columndatatype>
    <dim_operator>dim_operator0</dim_operator> LIKE
    <facttablecolumn>facttablecolumn0</facttablecolumn>
    <item_color>
    <item_shape>
    <item_row_number>
    <item_is_synonym>
</item>
</panel>

```

Element Name	Description
panel	A concept used to group items. The set of observation facts for each item filter are "unioned" at the panel level. The panel has the attribute, "name" which is the key field for the panel and it is unique.
panel_number	A serial number starting with 1.
panel_date_from	Apply the observation fact's start date condition at the panel level.
panel_date_to	Apply the observation fact's end date condition at the panel level.
invert	The invert value could be "1" or "0". If this value is "1", then query applies "NOT" condition for whole panel.
total_item_occurrences	Select the events only if the total number of occurrence is greater or equal to this value.
item	Item contains the filter and query building information, like the item key, dimension table column name, data type, etc.
hlevel	Hierarchy level not required for this implementation.
item_name	Name of the item, this is not required element and mostly for UI purposes.
item_table	Dimension table name.

item_key	Item key representing the unique path of concepts available in metadata schema or the ontology cell. The format of item_key is [\\Dimension\concept path].																				
item_icon	Not currently used.																				
tooltip	This is not required element. It is used by the user interface (UI).																				
class	This is not used, but just added to the specification. This is could be used to classify the data, for example whether we need fact's image data, text data, etc.																				
constrain_by_value	To constrain the observation value of a concept. More information can be found in the CRC design document.																				
value_operator	The conditional operator for filtering. The values are: 'EQ', 'NE', 'GT', 'GE', 'LT', 'LE', 'IN', 'BETWEEN', and 'LIKE'.																				
value_constraint	<p><u>Example 'GT' Operator:</u></p> <pre><constrain_by_value> <value_operator>GT</value_operator> <value_constraint>99.9</value_constraint> <value_type>NUMBER</value_type> </constrain_by_value></pre> <p><u>Example 'IN' Operator:</u></p> <pre><value_constraint>('NEG','NEGATIVE')</value_constraint></pre> <p><u>Example 'BETWEEN' Operator:</u></p> <pre><value_constraint>100 and 200</value_constraint></pre>																				
value_unit_of_measure																					
value_type	<p>TEXT, NUMBER, FLAG</p> <p>The following shows how the sql will be built for the different operators:</p> <table><tr><th>Value_type</th><th>Valuetype_cd</th><th>tval_char</th><th>Valueflag_cd</th><th>nval_num</th></tr><tr><td>TEXT</td><td>T</td><td>?</td><td></td><td></td></tr><tr><td>NUMBER</td><td>N</td><td>?</td><td></td><td>?</td></tr><tr><td>FLAG</td><td></td><td></td><td>?</td><td></td></tr></table>	Value_type	Valuetype_cd	tval_char	Valueflag_cd	nval_num	TEXT	T	?			NUMBER	N	?		?	FLAG			?	
Value_type	Valuetype_cd	tval_char	Valueflag_cd	nval_num																	
TEXT	T	?																			
NUMBER	N	?		?																	
FLAG			?																		
constrain_by_date	Apply start and end date constraint for the item.																				

<i>date_from</i>	<date_from time="start_date/end_date" inclusive="yes"></date_from>
<i>date_to</i>	<date_to time="start_date/end_date" inclusive="yes"></date_to>
dim_tablename	The name of the dimension table to join with the fact table. i.e. 'concept_dimension', 'provider_dimension', etc. This information is used to construct the dimension filter SQL. <i>Example:</i> select * from observation_fact where facttablecolumn in (select dim_columnname from dim_tablename where dim_columnname like dim_dimcode)
dim_columnname	The column name of the dimension table.
dim_dimcode	This is same as the concept path. i.e. '\i2b2\Diagnoses'
dim_columndatatype	The data type of dimension table's filter column. The default is string.
dim_operator	The conditional operator for filtering. The default is the 'LIKE' operator. Other values are 'LE', 'GE', and 'EQ'.
facttablecolumn	This is name of the column in the observation fact table to join the dimension table.
item_color	UI rendering attribute.
item_shape	UI rendering attribute.
item_row_number	UI rendering attribute.
item_is_synonym	UI rendering attribute.

2.3.2.1 REQUEST AND RESPONSE MESSAGE STRUCTURE

The patient set request/response message structure is divided into three parts:

1. PSMHeader
2. Request
3. Response

For a request message, the <psmheader> and <request> parts are required; while a response message requires only the <response> part.

```

<i2b2:message_body>
  <crc:psmheader>
    <request_type></request_type>
  </crc:psmheader>
  <crc:request>

```

```

...
</crc:request>
<crc:response>
...
</crc:response>
</i2b2:message_body>

```

Element name	Description
<psmheader>	<p>The header has a <request_type> element, which will carry the operation name. Each operation name has a specific request/response combination.</p> <p>Following is a list of supported operation names:</p> <ul style="list-style-type: none"> ▪ CRC_QRY_getRequestXml_fromQueryMasterId ▪ CRC_QRY_getQueryMasterList_fromUserId ▪ CRC_QRY_runQueryInstance_fromQueryDefinition ▪ CRC_QRY_getQueryMasterList_fromGroupId ▪ CRC_QRY_getQueryResultInstanceList_fromQueryInstanceId ▪ CRC_QRY_getQueryInstanceList_fromQueryMasterId ▪ CRC_QRY_deleteQueryMaster ▪ CRC_QRY_renameQueryMaster ▪ CRC_QRY_runQueryInstance_fromQueryMasterId ▪ CRC_QRY_getResultDocument_fromResultInstanceId ▪ CRC_QRY_getResultType ▪ CRC_QRY_runQueryInstance_fromAnalysisDefinition ▪ CRC_QRY_cancelQuery ▪ CRC_QRY_updateResultInstanceDescription
<request>	<p>The <request> is modeled as an object using a polymorphic approach. All operation specific request objects inherit a base RequestType object containing a request_type attribute as shown in the object model diagram in the next section.</p>
<response>	<p>The <response> is also modeled as an object using a polymorphic approach. All operation specific response objects inherit a base ResponseType object containing a StatusType attribute as shown in the object model diagram in the next section.</p>

2.3.3 Request and Response Object Model



The following chart shows the different request and response types for each service type listed above. The RequestType column describes what input is expected and the ResponseType column describes what output is expected.

Operation	RequestType						ResponseType					
	User	Master (Query)	Instance (Query Run)	QueryDefinition	PatientSet	ObservationFact	Master (Query)	Instance (Query Run)	Result	RequestXml	MasterInstanceResult	PatientData
Get a List of Saved Query Definitions	x						x					
Get a List of Saved Query Runs		x						x				
Get a List of Saved Query Results			x						x			
Get XML Definition of a Defined Query		x								x		
Run (New) Patient Set Query				x							x	
Run (Existing) Patient Set Query		x									x	
Get Patient Data From a Patient Set					x							x
Get Patient Data From Observation Fact						x						x

2.3.4 Use Case Scenarios

2.3.4.1 EXECUTE A QUERY AND GET ITS RESULTS

The service query's the data mart using the query definition and generates output based on the result option. Each query request and its results will be recorded under the given user id and project id.

The server will read the value <result_waittime_ms> from the <request_header> and if the query did not complete before the wait time specified in the request, it will send a response to the client with "PENDING" status. The client can later send a query instance request to see if the query is completed and get the query result information.

The service supports following result options.

- PATIENTSET
- PATIENT_COUNT_XML

- PATIENT_GENDER_COUNT_XML
- PATIENT_AGE_COUNT_XML
- PATIENT_VITALSTATUS_COUNT_XML
- PATIENT_RACE_COUNT_XML

If the result option (<result_output_list>) is not specified in the request, then the default result option is "PATIENTSET".

Note : To speed up the query, the CRC selects which panel's sql to run first. The selection of panel will be based on the individual item/concepts's total_num value specified on the Ontology's Metadata table. It is not necessary to populate total_num for each of the Metadata's concepts, but just populating some of the general concept like the '\Demographics\Gender\Male' would help.

Result Output Name	Description	Output Value
PATIENTSET	The patient set from the query will be persisted in the database	Not Applicable
PATIENT_COUNT_XML	Returns the patient count in i2b2 result xml format. The i2b2 result format is similar to the name-value pair.(i2b2_result_msg.xsd)	<pre> <i2b2_result_envelope xmlns:ns10="http://www.i2b2.org/xsd/hive/ msg/result/1.1/"> <body> <result name="PATIENT_COUNT_XML"> <data type="int" column="patient_count">1000 </data> </result> </body> </i2b2_result_envelope> </pre>
PATIENT_GENDER_COUNT_XML	Returns patient gender count in i2b2 result xml format. Refer i2b2_result_msg.xsd for i2b2 result format.	<pre> <i2b2_result_envelope xmlns:ns10="http://www.i2b2.org/xsd/hive/ msg/result/1.1/"> <body> <result name="PATIENT_GENDER_COUNT_XML"> <data type="int" column="Female">100</data> <data type="int" column="Male">200</data> <data type="int" column="Unknown">9</data> </result> </body> </i2b2_result_envelope> </pre>

		</body> </i2b2_result_envelope>
PATIENT_AGE_COUNT_XML	Returns patient age count in i2b2 result xml format. Refer i2b2_result_msg.xsd for i2b2 result format.	<pre> <i2b2_result_envelope xmlns:ns10="http://www.i2b2.org/xsd/hive/ msg/result/1.1/"> <body> <result name="PATIENT_AGE_COUNT_XML" > <data type="int" column=">= 65 years old">83</data> <data type="int" column=">= 85 years old">6</data> <data type="int" column="0-9 years old">1</data> <data type="int" column="10- 17 years old">0</data> <data type="int" column="18- 34 years old">16</data> <data type="int" column="35- 44 years old">31</data> <data type="int" column="45- 54 years old">32</data> <data type="int" column="55- 64 years old">44</data> <data type="int" column="65- 74 years old">42</data> <data type="int" column="75- 84 years old">35</data> <data type="int" column="zz not recorded">0</data> </ result> </body> </i2b2_result_envelope> </pre>
PATIENT_VITALSTATUS_COUNT_XML	Returns patient vital status count in i2b2 result xml format. Refer i2b2_result_msg.xsd for i2b2 result format.	<pre> <i2b2_result_envelope xmlns:ns10="http://www.i2b2.org/xsd/hive/ msg/result/1.1/"> <body> <result name="PATIENT_VITALSTATUS_CO UNT_XML"> <data type="int" Column="Deceased">0</data> <data type="int" column="Deferred">0</data> <data type="int" column="Living">123</data> </result> </body> </i2b2_result_envelope> </pre>

		<pre> <data type="int" column="Not recorded">0</data> </result> </body> </i2b2_result_envelope> </pre>
PATIENT_RACE_COUNT_XML	Returns patient race count in i2b2 result xml format. Refer i2b2_result_msg.xsd for i2b2 result format.	<pre> <i2b2_result_envelope xmlns:ns10="http://www.i2b2.org/xsd/hive /msg/result/1.1/"> <body> <result name="PATIENT_RACE_COUNT_XM L"> <data type="int" column="American Indian">30</data> <data type="int" column="Asian">40</data> <data type="int" column="Black">18</data> <data type="int" column="Hispanic">19</data> <data type="int" column="Indian">2</data> <data type="int" column="Not recorded">1866</data> <data type="int" column="White">100</data> </result> </body> </i2b2_result_envelope> </pre>

2.3.4.2 MESSAGE REQUEST AND RESPONSE

Request Type	Request	Response
CRC_QRY_runQueryInstance_fromQueryDefinition	query_definition_requestType	master_instance_result_responseType

Example :

```

<request_header>
  <result_waittime_ms>90000</result_waittime_ms>
</request_header>

<message_body>
  <crc:psmheader>
    <request_type>
      CRC_QRY_runQueryInstance_fromQueryDefinition
    </request_type>
  </crc:psmheader>

  <crc:request xsi:type="crc:query_definition_requestType">
    <query_definition>
      <query_name/>
      <query_description/>
      <panel>
        <panel_number>1</panel_number>
        <panel_date_from>2000-12-30T00:00:00</panel_date_from>
        <panel_date_to>2000-12-30T00:00:00</panel_date_to>
        <invert>0</invert>
        <total_item_occurrences>1</total_item_occurrences>
        <item>
          <item_key>\\rpd\RPDR\Diagnoses</item_key>
        </item>
      </panel>
    </query_definition>
    <result_output_list>
      <result_output name="PATIENT_COUNT_XML"/>
      <result_output name="PATIENT_GENDER_COUNT_XML "/>
      <result_output name="PATIENT_AGE_COUNT_XML "/>
      <result_output name="PATIENT_VITALSTATUS_COUNT_XML "/>
      <result_output name="PATIENT_RACE_COUNT_XML "/>
      <result_output name="PATIENTSET"/>
    </result_output_list>
  </crc:request>

  <crc:response xsi:type="crc:master_instance_result_responseType">
    <query_master>
      <query_master_id>0</query_master_id>
      <name/>
      <user_id/>
      <group_id/>
      <create_date>2000-12-30T00:00:00</create_date>
      <request_xml/>
    </query_master>
  </crc:response>

```

```

<query_instance>
  <query_instance_id>0</query_instance_id>
  <query_master_id>0</query_master_id>
  <user_id/>
  <group_id/>
  <batch_mode/>
  <start_date>2000-12-30T00:00:00</start_date>
  <end_date>2000-12-30T00:00:00</end_date>
  <query_status_type>
    <status_type_id>6</status_type_id>
    <name>COMPLETED</name>
    <description/>
  </query_status_type>
</query_instance>
<query_result_instance>
  <result_instance_id>0</result_instance_id>
  <query_instance_id>0</query_instance_id>
  <query_result_type>
    <result_type_id>1</result_type_id>
    <name>PATIENTSET</name>
    <description/>
  </query_result_type>
  <set_size>0</set_size>
  <obfuscate_method>OBTOTAL</obfuscate_method>
  <start_date>2000-12-30T00:00:00</start_date>
  <end_date>2000-12-30T00:00:00</end_date>
  <query_status_type>
    <status_type_id>3</status_type_id>
    <name>FINISHED</name>
    <description/>
  </query_status_type>
</query_result_instance>
<query_result_instance>
  <result_instance_id>0</result_instance_id>
  <query_instance_id>0</query_instance_id>
  <query_result_type>
    <result_type_id>4</result_type_id>
    <name> PATIENT_COUNT_XML </name>
    <description/>
  </query_result_type>
  <set_size>0</set_size>
  <obfuscate_method>OBSUBTOTAL</obfuscate_method>
  <start_date>2000-12-30T00:00:00</start_date>
  <end_date>2000-12-30T00:00:00</end_date>
  <query_status_type>

```

```

        <status_type_id>3</status_type_id>
        <name>FINISHED</name>
        <description/>
    </query_status_type>
</query_result_instance>
</crc:response>
</message_body>

```

2.3.4.3 SCENARIO: CHECK IF THE QUERY IS COMPLETED AND GET ITS RESULTS

This request accepts a query_instance_id and returns query status information. i.e. COMPLETED, RUNNING, ERROR, etc.

Request Type	Request	Response
CRC_QRY_getQueryResultInstanceList_fromQueryInstanceId	instance_requestType	result_responseType

Example:

```

<message_body>
  <psmheader>
    <user login="demo">demo</user>
    <patient_set_limit>0</patient_set_limit>
    <estimated_time>0</estimated_time>
    <request_type>
      CRC_QRY_getQueryResultInstanceList_fromQueryInstanceId
    </request_type>
  </psmheader>

  <request xsi:type="ns4:instance_requestType"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <query_instance_id>6280</query_instance_id>
  </request>

  <response xsi:type="ns5:result_responseType"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
    <query_result_instance>
      <result_instance_id>6280</result_instance_id>
    </query_result_instance>
  </response>
</message_body>

```

```

    <query_instance_id>6280</query_instance_id>
    <query_result_type>
      <result_type_id>1</result_type_id>
      <name>PATIENTSET</name>
    </query_result_type>
    <set_size>2000</set_size>
    <start_date>2007-09-06T10:42:14.000-04:00</start_date>
    <end_date>2007-09-06T10:42:15.000-04:00</end_date>
    <query_status_type>
      <status_type_id>3</status_type_id>
      <name>FINISHED</name>
    </query_status_type>
  </query_result_instance>
</response>
</message_body>

```

2.3.4.4 SCENARIO: GET THE QUERY RESULT (XML OUTPUT) BY RESULT INSTANCE

Pass the query result instance id and get the query result. If the result instance has a xml output, then the xml output will be passed in the <xml_value> and the <xml_value> element will contain an i2b2 result xml document.

Request Type	Request	Response
CRC_QRY_getResultDocument_fromResultInstanceId	result_requestType	crc_xml_result_responseType

Example:

```

<message_body>
  <psmheader>
    <request_type>
      CRC_QRY_getResultDocument_fromResultInstanceId
    </request_type>
  </psmheader>

  <ns4:request xsi:type="ns4:result_requestType"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <query_result_instance_id>7953</query_result_instance_id>
  </ns4:request>

```

```

<ns4:response xsi:type="ns4:crc_xml_result_responseType">
  <status>
    <condition type="DONE">DONE</condition>
  </status>
  <query_result_instance>
    <result_instance_id>7953</result_instance_id>
    <query_instance_id>8192</query_instance_id>
    <query_result_type>
      <result_type_id>4</result_type_id>
      <name>PATIENT_GENDER_COUNT_XML</name>
      <description>PATIENT GENDER COUNT XML</description>
    </query_result_type>
    <set_size>5815</set_size>
    <start_date>2008-07-02T11:07:05.000-04:00</start_date>
    <end_date>2008-07-02T11:07:05.000-04:00</end_date>
    <query_status_type>
      <status_type_id>3</status_type_id>
      <name>FINISHED</name>
      <description>FINISHED</description>
    </query_status_type>
  </query_result_instance>
  <crc_xml_result>
    <xml_result_id>804</xml_result_id>
    <result_instance_id>7953</result_instance_id>
    <xml_value><?xml version="1.0" encoding="UTF-8" standalone="yes"?>
      <ns9:i2b2_result_envelope
        xmlns:ns9="http://www.i2b2.org/xsd/hive/msg/result/1.1/">
        <body>
          <ns9:result name="patient_gender_count">
            <data type="int" column="male_count">1559</data>
            <data type="int" column="female_count">4256</data>
          </ns9:result>
        </body>
      </ns9:i2b2_result_envelope>
    </xml_value>
  </crc_xml_result>
</ns4:response>
</message_body>

```


2.3.4.5 SCENARIO: RUN AN ANALYSIS PLUG-IN AND GET THE STATUS

Run an Analysis plug-in by passing the plug-in name and its parameters. The analysis name and parameter are usually part of the individual Analysis plug-in document. The response message for this request is similar to the Setfinder's run query request.

Request Type	Request	Response
CRC_QRY_runQueryInstance_fromAnalysisDefinition	analysis_definitionType	master_instance_result _responseType

Example:

```
<request_header>
  <result_waittime_ms>90000</result_waittime_ms>
</request_header>

<message_body>
  <crc:psmheader>
    <request_type>
      CRC_QRY_runQueryInstance_fromAnalysisDefinition
    </request_type>
  </crc:psmheader>

  <crc:request xsi:type="crc:analysis_definition_requestType">
    <analysis_definition>
      <analysis_plugin_name>CALCULATE_PATIENTCOUNT_FROM_CONCEPTPATH
    </analysis_plugin_name>
    <crc_analysis_input_param name="ONT request">
      <param type="int" column="item_key">
        \\rpdr\RPDR\Diagnoses\Circulatory system (390-459)\
      </param>
    </crc_analysis_input_param>
    <crc_analysis_result_list>
      <result_output full_name="XML" priority_index="1" name="XML"/>
    </crc_analysis_result_list>
  </analysis_definition>
</crc:request>

  <crc:response xsi:type="crc:master_instance_result_responseType">
    <query_master>
```

```

    <query_master_id>0</query_master_id>
    <name>CALCULATE_PATIENTCOUNT_FROM_CONCEPTPATH</name>
    <user_id/>
    <group_id/>
    <create_date>2000-12-30T00:00:00</create_date>
    <request_xml/>
  </query_master>
  <query_instance>
    <query_instance_id>0</query_instance_id>
    <query_master_id>0</query_master_id>
    <user_id/>
    <group_id/>
    <batch_mode/>
    <start_date>2000-12-30T00:00:00</start_date>
    <end_date>2000-12-30T00:00:00</end_date>
    <query_status_type>
      <status_type_id>6</status_type_id>
      <name>COMPLETED</name>
      <description/>
    </query_status_type>
  </query_instance>
  <query_result_instance>
    <result_instance_id>0</result_instance_id>
    <query_instance_id>0</query_instance_id>
    <query_result_type>
      <query_result_type>
        <result_type_id>3</result_type_id>
        <name>XML</name>
        <display_type>CATNUM</display_type>
        <visual_attribute_type>LH</visual_attribute_type>
        <description>Generic query result</description>
      </query_result_type>
      <set_size>0</set_size>
      <obfuscate_method/>
      <start_date>2000-12-30T00:00:00</start_date>
      <end_date>2000-12-30T00:00:00</end_date>
      <query_status_type>
        <status_type_id>6</status_type_id>
        <name>COMPLETED</name>
        <description>COMPLETED</description>
      </query_status_type>
    </query_result_instance>
  </crc:response>
</message_body>

```

2.3.4.6 SCENARIO: GET A LIST OF QUERIES BY USER ID

This request fetches a list of query master information for the given user id. The client can also specify how many query master items to return from the server using the <fetch_size> element. The server returns query master items in descending order of query creation time.

Request Type	Request	Response
CRC_QRY_getQueryMasterList_fromUserId	user_requestType	master_responseType

Example:

```
<message_body>
  <psmheader>
    <request_type>
      CRC_QRY_getQueryMasterList_fromUserId
    </request_type>
  </psmheader>

  <request xsi:type="ns3:user_requestType"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <user_id>user1</user_id>
    <fetch_size>100</fetch_size>
  </request>

  <response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:type="ns5:master_responseType">
    <status>
      <condition type="DONE">DONE</condition>
    </status>
    <query_master>
      <query_master_id>6302</query_master_id>
      <name>1 y-Femal-Rheum@10:17:55</name>
      <user_id>demo</user_id>
      <group_id>Asthma</group_id>
      <create_date>2007-09-06T22:17:57.000-04:00</create_date>
    </query_master>
  </response>
```

```

        <query_master_id>6301</query_master_id>
        <name>10 ye-Female@10:42:41</name>
        <user_id>demo</user_id>
        <group_id>Asthma</group_id>
        <create_date>2007-09-06T10:42:42.000-04:00</create_date>
    </query_master>
</response>
</message_body>

```

2.3.4.1 SCENARIO: GET METADATA OF ANALYSIS PLUGIN

To return the analysis plugin's metadata by <plugin_name>. The <plugin_name> value of "ALL" is used to return all the plugin's metadata.

Request Type	Request	Response
CRC_QRY_getAnalysisPluginMetadata	analysis_plugin_metadata_requestType	analysis_plugin_metadata_responseType

Example:

```

<request_header>
<result_waittime_ms>90000</result_waittime_ms>
</request_header>

<message_body>
<crc:psmheader>
<request_type>
CRC_QRY_getAnalysisPluginMetadata
</request_type>
</crc:psmheader>

<crc:request xsi:type="crc:analysis_plugin_metadata_requestType ">
<plugin_name>ALL</plugin_name>
</crc:request>

<crc:response xsi:type="crc:analysis_plugin_metadata_responseType">
    <analysis_plugin_metadata_type>
    <plugin_id>1</plugin_id>
    <plugin_name>
        CALCULATE_PATIENTCOUNT_FROM_CONCEPTPATH
    </plugin_name>

```

```

<description>Plugin description</description>
<version_cd>1.0</version_cd>
<parameter_info>
  <crc_analysis_input_param name="name0">
    <param column="item_key" type="string">some value</param>
  </crc_analysis_input_param>
</parameter_info>
<parameter_info_xsd />
<status_cd>A</status_cd>
<command_line>
  /opt/jboss/server/default/analysis_commons_launcher/bin/co.sh
</command_line>
<working_folder>
  /opt/jboss/server/default/analysis_commons_launcher/bin
</working_folder>
<group_id>project_id</group_id>
</analysis_plugin_metadata_type>
</crc:response>

```

2.3.4.2 SCENARIO: GET QUERY DEFINITION FROM MASTER ID

This request will return <query_definition> information for the given query master id.

Request Type	Request	Response
CRC_QRY_getRequestXml_fromQueryMasterId	master_requestType	request_xml_responseType

Example:

```

<message_body>
  <psmheader>
    <request_type>
      CRC_QRY_getRequestXml_fromQueryMasterId
    </request_type>
  </psmheader>

  <request xsi:type="ns4:master_requestType"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <query_master_id>6300</query_master_id>
  </request>

```

```

<response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:type="request_xml_responseType ">
  <status>
    <condition type="DONE">DONE</condition>
  </status>
  <request_xml><![CDATA[
    <query_definition>
      <query_name/>
      <query_description/>
      <query_timing>ANY</query_timing>
      <specificity_scale>0</specificity_scale>
      <query_date_from>2000-12-30T00:00:00</query_date_from>
      <query_date_to>2000-12-30T00:00:00</query_date_to>
      <panel>
        <panel_number>0</panel_number>
        <panel_date_from>2000-12-30T00:00:00</panel_date_from>
        <panel_date_to>2000-12-30T00:00:00</panel_date_to>
        <invert>0</invert>
        <total_item_occurrences>0</total_item_occurrences>
        <item>
          <hlevel>0</hlevel>
          <item_name/>
          <item_table/>
          <item_key/>
          <item_icon/>
          <tooltip/>
          <class/>
        </item>
      </panel>
    </query_definition>]]>
  </request_xml>
</response>
</message_body>

```

2.3.4.3 SCENARIO: RENAME A QUERY

Use this request to change the name of the query. If the same user already has a query with the specified name, then the server will return an error in the <status> tag.

Request Type	Request	Response
CRC_QRY_renameQueryMaster	master_rename_requestType	master_responseType

Example:

```
<message_body>
  <psmheader>
    <user_login="demo">demo</user>
    <patient_set_limit>0</patient_set_limit>
    <estimated_time>0</estimated_time>
    <request_type>CRC_QRY_renameQueryMaster</request_type>
  </psmheader>

  <request xsi:type="ns4:master_rename_requestType"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <user_id>demo</user_id>
    <query_master_id>5997</query_master_id>
    <query_name>Demographics@03:21:10 -n[07-20-2007 ]</query_name>
  </request>

  <response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:type="ns5:master_responseType">
    <status>
      <condition type="DONE">DONE</condition>
    </status>
    <query_master>
      <query_master_id>5997</query_master_id>
      <name>Demographics@03:21:10 -n[07-20-2007 ]</name>
      <user_id>demo</user_id>
    </query_master>
  </response>
</message_body>
```

2.3.4.4 SCENARIO: DELETE A QUERY

Use this request to remove a query and its results. Delete will not permanently remove the query; it will just set the delete flag to true.

Request Type	Request	Response
CRC_QRY_deleteQueryMaster	master_delete_requestType	master_responseType

Example:

```
<message_body>
  <psmheader>
    <user_login="demo">demo</user>
    <patient_set_limit>0</patient_set_limit>
    <estimated_time>0</estimated_time>
    <request_type>CRC_QRY_deleteQueryMaster</request_type>
  </psmheader>

  <request xsi:type="ns4:master_delete_requestType"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <user_id>demo</user_id>
    <query_master_id>5997</query_master_id>
  </request>

  <response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:type="ns5:master_responseType">
    <status>
      <condition type="DONE">DONE</condition>
    </status>
    <query_master>
      <query_master_id>5997</query_master_id>
    </query_master>
  </response>
</message_body>
```

2.3.4.5 SCENARIO: CANCEL A QUERY

The process of cancelling a query simply stops it from moving to the next queue. It does not stop the execution if the query has already started.

Request Type	Request	Response
CRC_QRY_cancelQuery	instance_requestType	Instance_result_reponseType

Example:


```

<message_body>
  <psmheader>
    <request_type>CRC_QRY_cancelQuery</request_type>
  </psmheader>
  <request xsi:type="ns4:instance_requestType">
    <query_instance_id>1013</query_instance_id>
  </request>
  <crc:response xsi:type="crc:instance_result_responseType">
    <query_instance>
      <query_instance_id>0</query_instance_id>
      <query_master_id>0</query_master_id>
      <user_id/>
      <group_id/>
      <batch_mode/>
      <start_date>2000-12-30T00:00:00</start_date>
      <end_date>2000-12-30T00:00:00</end_date>
      <query_status_type>
        <status_type_id>9</status_type_id>
        <name>CANCELLED</name>
        <description/>
      </query_status_type>
    </query_instance>
    <query_result_instance>
      <result_instance_id>0</result_instance_id>
      <query_instance_id>0</query_instance_id>
      <query_result_type>
        <query_result_type>
          <result_type_id>3</result_type_id>
          <name>XML</name>
          <display_type>CATNUM</display_type>
          <visual_attribute_type>LH</visual_attribute_type>
          <description>Generic query result</description>
        </query_result_type>
        <set_size>0</set_size>
        <obfuscate_method/>
        <start_date>2000-12-30T00:00:00</start_date>
        <end_date>2000-12-30T00:00:00</end_date>
        <query_status_type>
          <status_type_id>9</status_type_id>
          <name>CANCELLED</name>
          <description>CANCELLED</description>
        </query_status_type>
      </query_result_instance>
    </crc:response>
  </message_body>

```

2.4 Patient Data Object Query Service

As the name suggests, these queries return patient data objects (PDO) in the response message as specified by the request message's patient set and the filter criteria. The message structure divided into three parts:

1. pdoheader
2. Request
3. Response

For the request message, the <pdoheader> and <request> parts are required, while for the response message, only the <response> part is required.

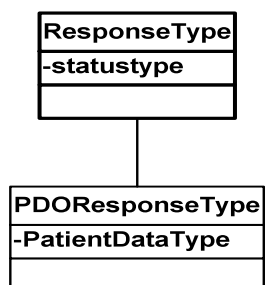
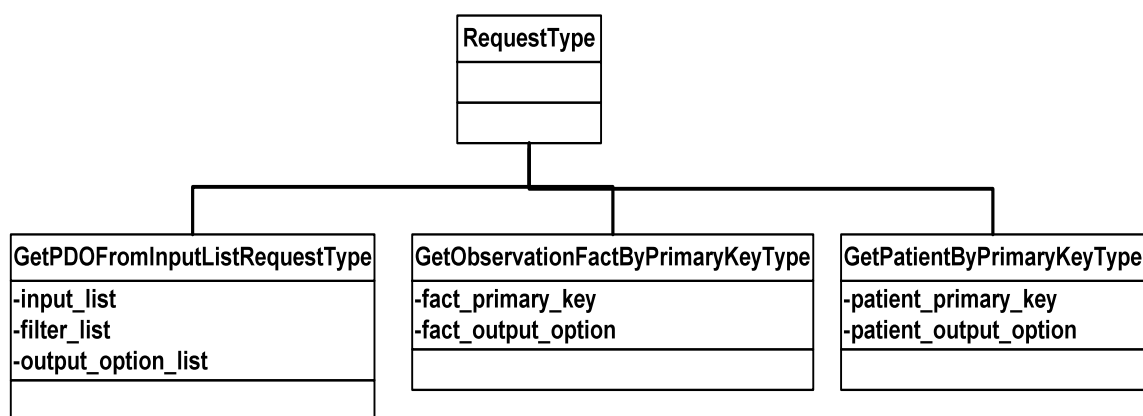
```
<i2b2:message_body>
  <crc:pdoheader>
    <request_type>GetPDOFromInputList_requestType</request_type>
    . . .
  </crc:pdoheader>
  <crc:request>
    . . .
  </crc:request>
  <crc:response>
    . . .
  </crc:response>
</i2b2:message_body>
```

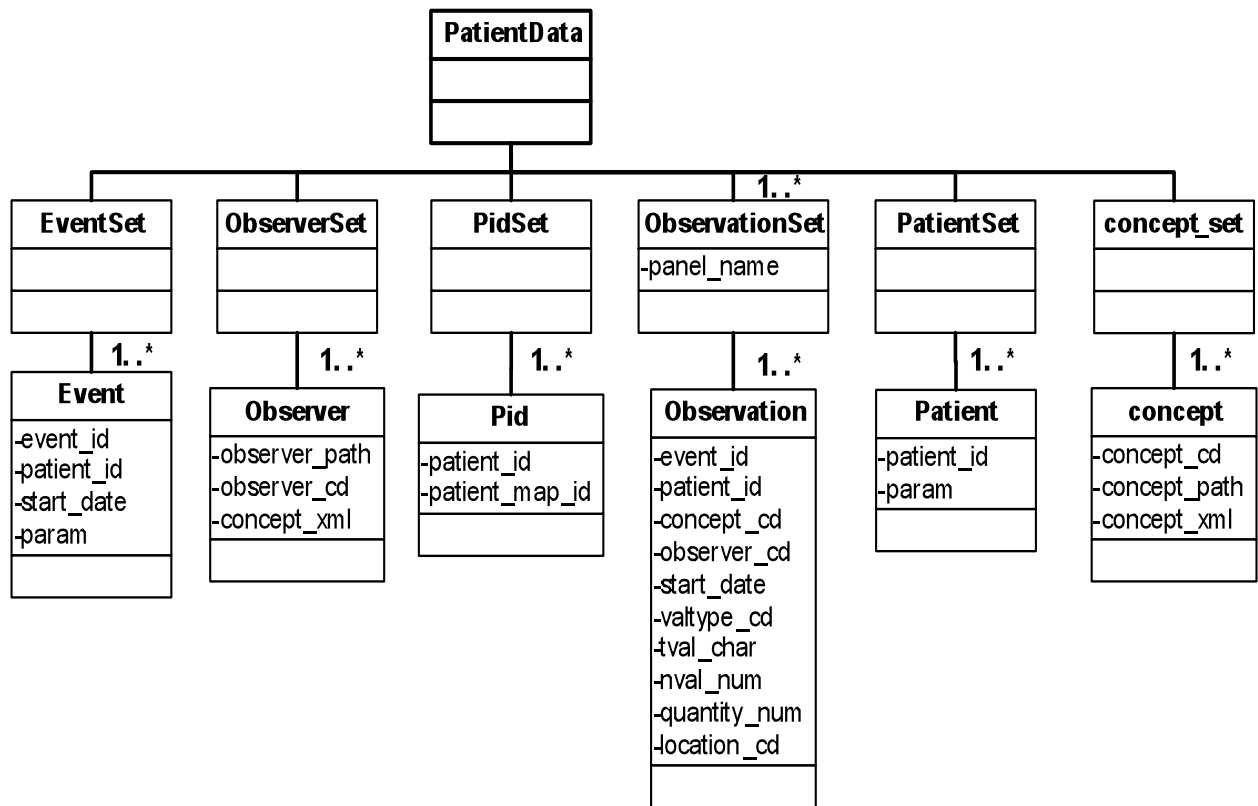
2.4.1 Request and Response message structure

Element name	Description
<pdoheader>	<p>The header contains a <request_type> element, which will carry the operation name. Each operation name has a specific request/response combination.</p> <p>Following are list of supported operation names:</p> <ul style="list-style-type: none">▪ getPDO_fromInputList▪ get_observationfact_by_primary_key▪ get_patient_by_primary_key

	<ul style="list-style-type: none"> ▪ <code>get_event_by_primary_key</code> ▪ <code>get_concept_by_primary_key</code> ▪ <code>get_observer_by_primary_key</code>
<request>	The <request> is modeled as an object using a polymorphic approach. All operation specific request objects inherit a base RequestType object as shown in section 4.4.2.
<response>	The <response> is also modeled as an object using a polymorphic approach. All operation specific response objects inherit a base PDOResponseType object containing a StatusType attribute and a PatientData object as shown in the object model located in the next section.

2.4.2 Request and Response Object Model





2.4.3 Use Case Scenario

2.4.3.1 SCENARIO: GET PATIENT DATA FROM A PATIENT SET ID

This request divided into three parts:

1. input_list

The input_list accepts either the id of the patient set or a list of patient ids.

2. filter_list

The filter_list holds a list of panels. Panels in turn have item details that are used in constructing a PDO query.

3. output_option

The output_option specifies which set of patient data to return.

Each of the patient data sections in output_option has attributes to specify the level of detail data expected in the response.

Request Type	Request	Response
getPDO_fromInputList	GetPDOFromInputList_requestType	master_responseType

2.4.3.1.1 Input Option List Type

The input list to the PDO request can be either be in one of the following format. i.e. the patient_list , encounter_list , pid_list and eid_list.

Element Name	Description
patient_list	<p>The patient list input can be in, one of the three forms.</p> <ol style="list-style-type: none"> 1. patient set id(<patient_set_coll_id>) 2. the enumeration list of patient id(<patient_id>) 3. the entire patients of datamart(<entire_patient_set>) <p><i>Example for <patient_list>:</i></p> <pre><patient_list max="10" min="0"> <patient_id index="0">86850</patient_id> <!-- OR --> <patient_set_coll_id>4741</patient_set_coll_id> <!-- OR --> <entire_patient_set >true</entire_patient_set> </patient_list></pre>
encounter_list	<p>The encounter_list input format is similar to the patient_list. This list can be in one of the three forms.</p> <ol style="list-style-type: none"> 1. encounter set id (<encounter_set_coll_id>) 2. the encumeration list of encounter id(<encounter_id>) 3. the entire encounters of datamart(<entire_encounter_set>)

pid_list	<p>The pid_list consists of the list of enumerated patient_id and its source value and this input allows the user to query the data via the patient_id and its source. The PDO service first maps the patient_id to the patient_num before querying the datamart.</p> <p><u>Example for <pid_list>:</u></p> <pre> <pid_list> <pid index="1" source="MGH_E">PAT_MGH_001</pid> <pid index="2" source="BWH_E">PAT_BWH_001</pid> </pid_list> </pre>
eid_list	<p>The eid_list consists of the list of enumerated encounter_id and its source value and this input allows the user to query the data via the encounter_id and its source. The PDO service first maps the encounter_id to the encounter_num before querying the datamart.</p> <p><u>Example for <eid_list>:</u></p> <pre> <eid_list> <eid index="1" source="MGH_E">ENC_MGH_001</pid> <eid index="2" source="BWH_E">ENC_BWH_001</pid> </eid_list> </pre>

2.4.3.1.2 Filter List Type

Please refer to the earlier section titled *Panel Definition Detail* .

2.4.3.1.3 Output Option List Type

This option specifies the list of sections that is required in the PDO response.

Attribute Name	Description	Implemented
names="asattributes none"	This option specifies to return the names for the codes when returning the PDO data. The names for the codes are available in the code_lookup table.	Yes
phi="encrypted unencrypted"		No
time="nozone withzone"		No

Example:

```
<!-- output options -- >
<output_option names="asattributes">
  <patient_set select="using_filter_list" onlykeys="true"/>
  <concept_set select="using_filter_list" onlykeys="true"/>
  <observation_set blob="false" onlykeys="false"/>
  <event_set select="using_filter_list" onlykeys="true"/>
  <pid_set select="using_filter_list" onlykeys="true"/>
  <eid_set select="using_filter_list" onlykeys="true"/>
  <observer_set_using_filter_list onlykeys="true"/>
</output_option>
```

The following are the available options for each element in the <output_option>.

Attribute Name	Description
onlykeys = "true false"	If this option is "true", then only the primary keys will be returned. If it is "false", then all the fields except the blob and tech data fields will be returned.
blob="true false"	This option specifies whether the blob field is required in the output.
techdata = "true false"	This option specifies whether the tech data fields are required in the output. The following are a few examples of the tech data fields: sourcesystem_cd, import_date, update_date, etc.
select = "using_filter_list using_input_list"	There are a couple of ways to select the PDO data. If the select option is "using_input_list", then the returned data will be based on the <input_list> and the fact table will not be used to return the data. If it is "using_filter_list", then return the data that satisfies the <filter_list> constraints when applied on the fact table.

Element Name	Description
--------------	-------------

patient_set	<p>Return the set of patient dimension data either for the input list or from the patient present in the observation set.</p> <pre><output_option> <patient_set select="using_filter_list" onlykeys="true"/> </output_option></pre>																				
observation_set	<p>Return the observation set of the patient data object. There could be a multiple number of <observation_set> returned and the number of <observation_set> returned will be equal to number of panel defined in the filter list.</p> <p>Observation set has the attribute "panel_name" which corresponds to "name" attribute defined in the <panel>.</p>																				
event_set	<p>Return the set containing event/visit dimension data occurring in the observation set or from the input list.</p>																				
pid_set	<p>Return the patient mapping table's information for the given input list or from the observation set.</p> <table><tr><th>patient_id</th><th>patient_id_source</th><th>patient_num</th><th>patient_id_status</th></tr><tr><td>1000000</td><td>HIVE</td><td>1000000</td><td>A</td></tr><tr><td>1000001</td><td>HIVE</td><td>1000001</td><td>A</td></tr><tr><td>123</td><td>MGH</td><td>1000001</td><td>A</td></tr><tr><td>777</td><td>BWH</td><td>1000001</td><td>A</td></tr></table> <pre><pid_set> <!--pid with only the hive number - -> <pid> <patient_id status="A" source="HIVE">1000000</patient_id> </pid> <!-- - pid with hive number and mapping patient_id's - -> <pid> <patient_id status="A" source="HIVE">1000001</patient_id> <patient_map_id status="A" source="MGH">123</patient_map_id> <patient_map_id status="A" source="BWH">777</patient_map_id> </pid> </pid_set></pre>	patient_id	patient_id_source	patient_num	patient_id_status	1000000	HIVE	1000000	A	1000001	HIVE	1000001	A	123	MGH	1000001	A	777	BWH	1000001	A
patient_id	patient_id_source	patient_num	patient_id_status																		
1000000	HIVE	1000000	A																		
1000001	HIVE	1000001	A																		
123	MGH	1000001	A																		
777	BWH	1000001	A																		
eid_set	<p>Return the encounter mapping table's information for the given input list or from the</p>																				

	observation set.
observer_set_using_filter_list	Return the set containing observer/provider dimension data occurring in the observation set.
concept_set_using_filter_list	Return the set of concept dimension data occurring in the observation set.

Example:

```

<message_body>

  <crc:pdoheader>
    <request_type>getPDO_fromInputList</request_type>
  </crc:pdoheader>

  <crc:request xsi:type="ns2:GetPDOFromInputList_requestType"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

    <!-- input list -->
    <input_list>
      <patient_list max="300" min="1">
        <patient_set_coll_id>100</patient_set_coll_id>
      </patient_list>
      <!-- or -->
      <event_list/>
      <!-- or -->
      <pid_list min="1" max="3">
        <pid index="1" source="HIVE">9876</pid>
        <pid index="3" source="HIVE">1</pid>
      </pid_list>
      <!-- or -->
      <eid_list min="1" max="5">
        <eid index="1" source="HIVE">12222313</eid>
      </eid_list>
    </input_list>

    <!-- filter list -->
    <filter_list>
      <panel name="panel1">
        <panel_invert>0</panel_invert>
        <panel_accuracy_scale></panel_accuracy_scale>
        <panel_start_date><panel_start_date>
        <panel_end_date></panel_end_date>
      </panel>
    </filter_list>
  </crc:request>

```

```

<item>
  <item_name> </item_name>
  <item_key> </item_key>
  <item_icon> </item_icon>
  <item_tooltip></item_tooltip>
  <dim_tablename> </dim_tablename>
  <dim_columnname> </dim_columnname>
  <dim_dimcode></dim_dimcode>
  <dim_columndatatype> </dim_columndatatype>
  <dim_operator> </dim_operator>
  <facttablecolumn></facttablecolumn>
  <value_constraint>
    <value_type></value_type>
    <value_operator></value_operator>
    <value_unitofmeasure></value_unitofmeasure>
    <value></value>
  </value_constraint>
</item>
. . .
</panel>

<panel name="panel2">
  <panel_invert>0</panel_invert>
  <panel_accuracy_scale></panel_accuracy_scale>
  <panel_start_date><panel_start_date>
  <panel_end_date></panel_end_date>
  <item>
    <item_name> </item_name>
    <item_key> </item_key>
    <item_icon> </item_icon>
    <item_tooltip></item_tooltip>
    <dim_tablename> </dim_tablename>
    <dim_columnname> </dim_columnname>
    <dim_dimcode></dim_dimcode>
    <dim_columndatatype> </dim_columndatatype>
    <dim_operator> </dim_operator>
    <facttablecolumn></facttablecolumn>
    <value_constraint>
      <value_type></value_type>
      <value_operator></value_operator>
      <value_unitofmeasure></value_unitofmeasure>
      <value></value>
    </value_constraint>
  </item>
</panel>

```

```

. . .
</filter_list>

<!-- output options -- >
<output_option names="asattributes">
  <patient_set select="using_filter_list" onlykeys="true"/>
  <concept_set select="using_filter_list" onlykeys="true"/>
  <observation_set blob="false" onlykeys="false"/>
  <event_set select="using_filter_list" onlykeys="true"/>
  <pid_set select="using_filter_list" onlykeys="true"/>
  <eid_set select="using_filter_list" onlykeys="true"/>
  <observer_set_using_filter_list onlykeys="true"/>
<!-- To specify generalized dimension type -- >
  <dimension_set_using_filter_list dimensionname="dimension1"
    onlykeys="true"/>
</output_option>
</crc:request>

<!-- response begin -- >
<response>
  <patient_data>
    <!-- patient set section begins
    <patient_set>
      <patient>
        <patient_id>patient_id6</patient_id>
        <param column="vital_status_cd"
          name="date interpretation code">param3</param>
        <param column="birth_date"
          name="birthdate">param3</param>
        <param column="death_date"
          name="date of death">param3</param>
        <param column="sex_cd"
          name="gender">param3</param>
        <param column="age_in_years_num"
          name="age">param3</param>
        <param column="language_cd"
          name="language">param3</param>
        <param column="race_cd"
          name="race">param3</param>
        <param column="religion_cd"
          name="religion">param3</param>
        <param column="marital_status_cd"
          name="marital status">param3</param>
        <param column="statecityzip_path_char"
          name="zip code hierarchy">param3</param>

```

```

        </patient>
        . . .
    <patient_set>

    <!-- - concept set section begins -- >
    <concept_set>
        <concept>
            <concept_path>concept_path0</concept_path>
            <concept_cd>concept_cd0</concept_cd>
            <name_char>name_char0</name_char>
        </concept>
        . . .
    </concept_set>

    <!-- - observation set section begins -- >
    <observation_set panel_name="panel1">
        <observation>
            <event_id source="source3">event_id3</event_id>
            <patient_id>patient_id9</patient_id>
            <concept_cd name="name0">concept_cd3</concept_cd>
            <observer_cd source="source0">observer_cd3</observer_cd>
            <start_date>2006-05-04T18:13:51.0Z</start_date>
            <modifier_cd name="name1">modifier_cd0</modifier_cd>
            <valuetype_cd>valuetype_cd0</valuetype_cd>
            <tval_char>tval_char0</tval_char>
            <nval_num units="units0">3.141592653589</nval_num>
            <valueflag_cd name="name2">valueflag_cd0</valueflag_cd>
            <quantity_num>3.141592653589</quantity_num>
            <units_cd>units_cd0</units_cd>
            <end_date>2006-05-04T18:13:51.0Z</end_date>
            <location_cd name="name3">location_cd0</location_cd>
        </observation>
        <observation>
            . . .
        </observation>
    </observation_set>

    <observation_set panel_name="panel2">
        <observation>
            . . .
        </observation>
        <observation>
            . . .
        </observation>
        . . .
    </observation_set>

```

```

</observation_set>

<!-- event set section begins -- >
<event_set>
  <event>
    <event_id source="source0">event_id0</event_id>
    <patient_id>patient_id0</patient_id>
    <start_date>2006-05-04T18:13:51.0Z</start_date>
    <end_date>2006-05-04T18:13:51.0Z</end_date>
    <param column="inout_cd"
      name="in vs. outpatient code"></param>
    <param column="location_cd"
      name="location code"></param>
    <param column="location_path"
      name="location hierarchy"></param>
    <param column="active_status_cd"
      name="date interpretation code"></param>
  </event>
  . . .
</event_set>

<!-- observer/provider set section begins -- >
<observer_set>
  <observer>
    <observer_path>observer_path0</observer_path>
    <observer_cd>observer_cd0</observer_cd>
    <name_char>name_char3</name_char>
  </observer>
  . . .
</observer_set>

<!-- pid set section begins -- >
<pid_set>
  <pid>
    <patient_id status="A" source="HIVE">123456</patient_id>
  </pid>
  <pid>
    <patient_id status="A" source="HIVE">234567</patient_id>
    <patient_map_id status="A" source="EMP_E">ABC1
    </patient_map_id>
    <patient_map_id status="A" source="MGH_E">XYZ1
    </patient_map_id>
  </pid>
  . . .
</pid_set>

```

```

        </crc:patient_data>
    </response>
    <!-- response end -->
</message_body>

```

2.4.3.2 SCENARIO: GET OBSERVATION BLOB BY PRIMARY KEY

This request returns observation blob using observation primary key

Example:

```

<message_body>
  <pdoheader>
    <request_type>get_observationfact_by_primary_key</request_type>
  </pdoheader>

  <ns5:request
    xsi:type="ns5:GetObservationFactByPrimaryKey_requestType"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <fact_primary_key>
      <event_id>2004005981</event_id>
      <patient_id>52003</patient_id>
      <concept_cd>LCS-I2B2:c1009c</concept_cd>
      <observer_id>03840261</observer_id>
      <start_date>1995-08-24T00:00:00.179-05:00</start_date>
    </fact_primary_key>
    <fact_output_option select="using_filter_list" onlykeys="false"/>
  </ns5:request>

  <response>
    <patient_data>
      <observation_set>
        <observation>
          <event_id source="source3">event_id3</event_id>
          <patient_id>patient_id9</patient_id>
          <concept_cd name="name0">concept_cd3</concept_cd>
          <observer_cd source="source0">observer_cd3</observer_cd>
          <start_date>2006-05-04T18:13:51.0Z</start_date>
          <observation_blob><![CDATA[
            patient notes]]>
          </observation_blob>
        </observation>
      </observation_set>
    </patient_data>
  </response>

```

```

        </observation_set>
    </patient_data>
</response>
</message_body>

```

2.5 Data Upload Messages

The data loading process loads the data to three main sections of data mart:

1. Dimension tables
2. Mapping table
3. Observation fact table

 **Note:** In order to upload data a user must have the role of `DATA_DEID`.

Except for the `observation_fact` table, the data load will perform the insert or update operation based on two criteria: (1) whether the data exists in the respective tables and (2) whether the PDO's update date is greater than the existing data's update date.

For example the following table shows the condition for update and insert operation for the `visit_dimension` table.

Loading Visit_Dimension	Condition
UPDATE	INPUT.ENCOUNTER_NUM = OBSFACT.ENCOUNTER_NUM AND INPUT.PATIENT_NUM = OBSFACT.PATIENT_NUM AND INPUT.UPDATE_DATE > OBSFACT.UPDATE_DATE
INSERT	INPUT.ENCOUNTER_NUM != OBSFACT.ENCOUNTER_NUM OR INPUT.PATIENT_NUM != OBSFACT.PATIENT_NUM

The data load on the `observation_fact` table is slightly different compared to other tables and it can be divided into two cases.

Case 1: Refresh Observation Fact

This case assumes the user is trying to load a fresh set of observation facts and expects to delete any old observation facts which have matching encounter_num and patient_num.

Set append_flag="false" to support this case.

Case 2: Incremental Observation Fact

In this case the user has the option to just add new observation facts that may match the encounter_num and patient_num of an existing observation_fact entry. The user should make sure they are sending a unique observation fact, which will not result in duplicating an existing observation_fact entry.

Set append_flag="true" to support this case.

2.5.1 i2b2 Import: Append Flag

Assumption: the record(s) in the update file (new record) has the same primary key as a record(s) in the associated table (existing record).

Primary Key includes:

- Encounter number
- Patient number
- Concept code
- Start date
- Modifier code
- Observer code

Append Flag = True

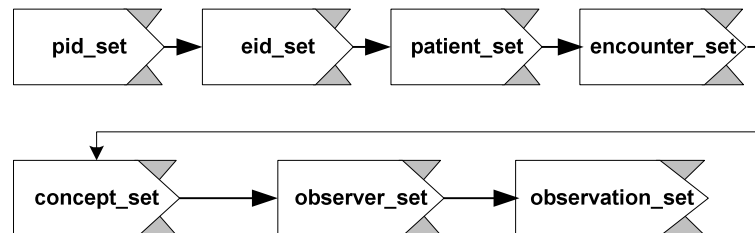
Following conditions will result in the new record **replacing** the existing record:

new record update date	equal to (=)		update date on the existing record	
new record update date	greater than (>)		update date on the existing record	
new record update date	is not null	AND	update date on the existing record	null
new record update date	null	AND	update date on the existing record	null

Following conditions will result in **ignoring** the new record and **not** updating the existing record:

new record update date	less than (<)		update date on the existing record	
new record update date	null	AND	update date on the existing record	is not null

The loader process loads the PDO data in a certain order, which is based on the data dependency of the data mart. The order of the data load is shown in the below diagram.



2.5.2 Use Case Scenarios

2.5.2.1 SCENARIO: UPLOAD DATA IN PATIENT DATA OBJECT XML

The data load request message can be divided into three sections.

1. The <input_list> specifies the location of the PDO document.
2. The <load_list> specifies the PDO sections to load.
3. The <output_list> specifies the output expected from the upload process.

In addition, certain dependencies exist for loading some sections of PDO. The following table shows the dependencies for loading the individual PDO section.

Load PDO section	Dependent PDO section
event_set	eid_set
patient_set	pid_set
observation_set	eid_set, pid_set, event_set, patient_set
concept_set, observer_set, eid_set, pid_set	NONE

For example, to load a single observation fact the required sections of the PDO XML are shown below.

 **NOTE: the dependency section in the PDO is optional; if they exist in the data mart.**

For more specific example on loading patient and encounter mapping, please refer to the *CRC Design* document.

```
<patient_data>
```

```
  <observation_set>
```

```
    <observation update_date="2006-05-04T18:13:51.0Z"
      sourcesystem_cd="TEST">
      <event_id source="MGHE">MENT001</event_id>
      <patient_id source="MGHP">MPAT001</patient_id>
      <concept_cd name="C">i2b2CD</concept_cd>
      <observer_cd source="TEST">@</observer_cd>
      <start_date>2006-05-04T18:13:51.0Z</start_date>
      <modifier_cd>modifier_cd</modifier_cd>
      <valuetype_cd>valuetype_cd0</valuetype_cd>
      <nval_num units="units0">3.141592653589</nval_num>
      <valueflag_cd name="name8">valueflag_cd0</valueflag_cd>
      <quantity_num>3.141592653589</quantity_num>
      <confidence_num>3.141592653589</confidence_num>
    </observation>
```

```
  </observation_set>
```

```
  <pid_set>
```

```

        <pid>
            <patient_id status="A" update_date="2006-05-04T18:13:51.0Z"
              sourcesystem_cd="TEST" source="MGHP">MPAT001</patient_id>
        </pid>
    </pid_set>

    <eid_set>
        <eid>
            <event_id source="MGHE" patient_id="MPAT001"
              patient_id_source="MGHP" status="A"
              update_date="2006-05-04T18:13:51.0Z"
              sourcesystem_cd="TEST">MENT001</event_id>
        </eid>
    </eid_set>

    <event_set>
        <event update_date="2006-05-04T18:13:51.0Z"
          sourcesystem_cd="TEST">
            <event_id source="MGHE">MENT001</event_id>
            <patient_id source="MGHP">MPAT001</patient_id>
            <start_date>2006-05-04T18:13:51.0Z</start_date>
            <end_date>2006-05-04T18:13:51.0Z</end_date>
        </event>
    </event_set>

</patient_data>

```

Request Message:

```

<message_body>
    <publish_data_request>
        <input_list>
            <data_file>
                <location_uri protocol_name="FR|LOCAL">location_uri0</location_uri>
                <data_format_type>PDO</data_format_type>
                <source_system_cd>source_system_cd0</source_system_cd>
                <load_label>load_label0</load_label>
            </data_file>
        </input_list>
        <load_list commit_flag="true" clear_temp_load_tables="false">
            <load_observation_set ignore_bad_data="true"/>
            <load_event_set ignore_bad_data="true"/>
        </load_list>
    </publish_data_request>
</message_body>

```

```

        <load_patient_set ignore_bad_data="true"/>
        <load_observer_set ignore_bad_data="true"/>
        <load_eventid_set ignore_bad_data="true"/>
        <load_pid_set ignore_bad_data="true"/>
        <load_eid_set ignore_bad_data="true"/>
        <load_concept_set encrypt_blob="false" ignore_bad_data="true"/>
    </load_list>
    <output_list detail="true">
        <observation_set onlykeys="true" blob="false" techdata="false"/>
        <patient_set onlykeys="true" blob="false" techdata="false"/>
        <event_set onlykeys="true" blob="false" techdata="false"/>
        <observer_set onlykeys="true" blob="false" techdata="false"/>
        <concept_set onlykeys="true" blob="false" techdata="false"/>
        <pid_set onlykeys="true" blob="false" techdata="false"/>
        <eid_set onlykeys="true" blob="false" techdata="false"/>
        <eventid_set onlykeys="true" blob="false" techdata="false"/>
    </output_list>
</publish_data_request>
</message_body>

```

<input_list>	This section of xml will carry input information such as the location of PDO file, its data format, etc.
location_uri	The location of the input PDO file.
protocol_name	Protocol name, which specifies the service how to access the PDO file in the location_uri. (FR – File Repository, LOCAL – File resides in the server local folder)
data_format_type	PDO (Patient Data Object XML)
source_system_cd	Upload's source system code
load_label	User's load label
<load_list>	This section of the request holds the load options.
commit_flag	The current implementation supports only "true" value. This is placeholder for future release. If this flag set to false, system will just run all the load process for the given PDO data and will not perform commit. This is useful option to check, whether the given PDO data, have the valid information
clear_temp_load_tables	This flag specifies whether to clean up staging area. Turn this flag false for debug purpose.
ignore_bad_data	Not implemented, will be part of future release.

load_observation <i>set, append_flag</i>	Load or update fact entry coming from input PDO data. If the append_flag = 'true' is specified then the PDO's fact information is just added to the stored fact and will not do any updates.
load_event_set	The information from the PDO's <event_set> is used to load to the visit_dimension table.
load_patient_set	The information from the PDO's <patient_set> is used to load to the patient_dimension table
load_observer_set <i>set delete_existing_data</i>	The information from the PDO's <observer_set> is used to load to the provider_dimension table. Set the delete_existing_flag to remove the old data before loading the new data and the old data will be available in the backup table.
load_pid_set	The information from the PDO's <pid_set> is used to load to the patient_mapping table.
load_eid_set	The information from the PDO's <eventid_set> is used to load to the encounter_mapping table.
load_concept_set <i>set delete_existing_data</i>	The information from the PDO's <concept_set> is used to load to the concept_dimension. Set the delete_existing_flag to remove the old data before loading the new data and the old data will be available in the backup table.
<output_list>	This section of the request is used for getting upload process status information.
detail	This attribute is a placeholder for a future release. The attribute specifies whether the response message should have detailed information. If the value is false, then the service will return just the count of inserted and updated records. Otherwise a separated file with inserted, updated records will be created.
observation_set	This element specifies the process to fetch the details of updated records on the observation_fact table.
onlykeys	Return primary key fields in the response
blob	Return blob field in the response
techdata	Return techdata field in the response
patient_set	The element specifies the process to fetch the details of updated records on the patient_dimension table
event_set	The element specifies the process to fetch the details of updated records on the visit_dimension table
observer_set	The element specifies the process to fetch the details of updated records on the provider_dimension table
concept_set	The element specifies the process to fetch the details of updated records on the concept_dimension table
pid_set	The element specifies the process to fetch the details of updated records on the patient_mapping table

eid_set	The element specifies the process to fetch the details of updated records on the encounter_mapping table
---------	--

Response Message:

```

<message_body>
  <load_data_response>
    <status>
      <condition type="ERROR|DONE" coding_system="">error message
    </condition>
    </status>
    <upload_id>upload_id0</upload_id>
    <user_id>user_id0</user_id>
    <data_file_location_uri protocol_name="FR|LOCAL"> location_uri0
    </data_file_location_uri>
    <load_status>load_status0</load_status>
    <transformer_name>transformer_name0</transformer_name>
    <start_date>2006-05-04T18:13:51.0Z</start_date>
    <end_date>2006-05-04T18:13:51.0Z</end_date>
    <message>message0</message>
    <observation_set inserted_record="0" ignored_record="0" total_record="0">
      <ignored_patient_data_file_uri protocol_name="FR|LOCAL"> uri0
      </ignored_patient_data_file_uri>
      <message>message1</message>
    </observation_set>
    <patient_set inserted_record="0" ignored_record="0" total_record="0">
      <ignored_patient_data_file_uri protocol_name="FR|LOCAL">
      </ignored_patient_data_file_uri>
      <message>message2</message>
    </patient_set>
    <event_set inserted_record="0" ignored_record="0" total_record="0">
    </event_set>
    <observer_set inserted_record="0" ignored_record="0" total_record="0">
    </observer_set>
    <concept_set inserted_record="0" ignored_record="0" total_record="0">
    </concept_set>
    <pid_set inserted_record="0" ignored_record="0" total_record="0">
    </pid_set>
    <eventid_set inserted_record="0" ignored_record="0" total_record="0">
    </eventid_set>
  </load_data_response>
</message_body>

```

2.5.2.2 SCENARIO: GET UPLOAD STATUS INFO BY UPLOAD_ID AND USER_ID

This message will retrieve one of the following:

- a list of upload status information based on user_id
- a single upload status by upload_id

Request Message:

```
<message_body>
  <get_upload_info_request>
    <user_id>user_id</user_id>
    <upload_id>100</upload_id>
  </get_upload_info_request>
</message_body>
```

Response Message:

```
<message_body>
  <load_data_list_response>
    <load_data_response>Please refer section 4.5.1.1 for <load_data_response/>
  </load_data_response>
  . . .
  <load_data_list_response>
</message_body>
```

2.6 Message Explanations

This section defines message elements in the CRC namespace (<http://www.i2b2.org/xsd/cell/crc/psm/1.1/> and <http://www.i2b2.org/xsd/cell/crc/pdo/1.1/>).

Each element defined will have an implied prefix of **crc:** unless another namespace is explicitly stated. Elements from other namespaces, which are included within

CRC elements, will be listed but not expanded or defined in this document. Refer to other cell documents to get specific details on those elements.

2.6.1 Header

The <header> is the first CRC element within an i2b2 <message_body>. This section defines the elements shown in the example header shown, below.

```
<crc:header>
  <user login=""></user>
  <data_source></data_source>
  <patient_set_limit>0</patient_set_limit>
  <estimated_time>0</estimated_time>
  <create_date>2002-12-23T00:00:00</create_date>
  <submit_date>2002-12-23T00:00:00</submit_date>
  <complete_date>2002-12-23T00:00:00</complete_date>
  <request_type>getPDO_fromObservationFact</request_type>
</crc:header>
```

header	container for generic information useful for any crc message
user	user information used for authentication and login
data_source	information about the source of the data
patient_set_limit	limit the size of the patient set returned in a query
estimated_time	the time estimated for the query to complete
create_date	the date that a query was created
submit_date	the date that a query was submitted to be executed or run
complete_date	the date that a query finished executing
request_type	a code that tells the service what type of request to expect, which tells it what kind of xml to expect in the rest of the message.

2.6.2 Requests

2.6.2.1 XSI:TYPE="CRC:USER_REQUESTTYPE"


```

<crc:request xsi:type="crc:user_requestType">
  <user_id>some_user_id</user_id>
  <group_id>some_group_id</group_id>
  <fetch_size>5000</fetch_size>
</crc:request>

```

2.6.2.2 XSI:TYPE="CRC:MASTER_REQUESTTYPE"

```

<crc:request xsi:type="crc:master_requestType">
  <query_master_id>0</query_master_id>
</crc:request>

```

2.6.2.3 XSI:TYPE="CRC:INSTANCE_REQUESTTYPE"

```

<crc:request xsi:type="crc:instance_requestType">
  <query_instance_id>0</query_instance_id>
</crc:request>

```

2.6.2.4 XSI:TYPE="CRC:QUERY_DEFINITION_REQUESTTYPE"

```

<crc:request xsi:type="crc:query_definition_requestType">
  <query_definition>
    <query_name/>
    <query_description/>
    <query_timing>SAME</query_timing>
    <specificity_scale>0</specificity_scale>
    <query_date_from>2000-12-30T00:00:00</query_date_from>
    <query_date_to>2000-12-30T00:00:00</query_date_to>
    <panel>
      <panel_number>0</panel_number>
      <panel_date_from>2000-12-30T00:00:00</panel_date_from>
      <panel_date_to>2000-12-30T00:00:00</panel_date_to>
      <invert>0</invert>
      <total_item_occurrences>0</total_item_occurrences>
      <item>
        <hlevel>0</hlevel>

```

```

        <item_name/>
        <item_table/>
        <item_key/>
        <item_icon/>
        <tooltip/>
        <class/>
    </item>
</panel>
</query_definition>
</crc:request>

```

2.6.2.5 XSI:TYPE="CRC:GETPDO_FROMINPUTLIST "

```

<crc:request xsi:type="crc:patient_set_requestType">
  <select_option_list>
    <observation_fact blob="true" before="2005-12-30T00:00:00"
      after="2003-12-30T00:00:00"/>
    <patient_dimension fact_related="false"/>
    <provider_dimension/>
    <visit_dimension detail="false"/>
    <concept_dimension status="true"/>
  </select_option_list>
  <filter_list>
    <panel name="panel1">
      <panel_invert>0</panel_invert>
      <panel_accuracy_scale></panel_accuracy_scale>
      <panel_start_date><panel_start_date>
      <panel_end_date></panel_end_date>
      <item>
        <item_name> </item_name>
        <item_key> </item_key>
        <item_icon> </item_icon>
        <item_tooltip></item_tooltip>
        <dim_tablename> </dim_tablename>
        <dim_columnname> </dim_columnname>
        <dim_dimcode></dim_dimcode>
        <dim_columndatatype> </dim_columndatatype>
        <dim_operator> </dim_operator>
        <facttablecolumn></facttablecolumn>
        <value_constraint>
          <value_type></value_type>
          <value_operator></value_operator>

```

```

        <value_unitofmeasure></value_unitofmeasure>
        <value></value>
    </value_constraint>
</item>
. . .
</panel>
</filter_list>
<patient_list min="1" max="10">
    <patient_num index="1">50</patient_num>
    <patient_num index="2">24</patient_num>
    <patient_num index="3">78</patient_num>
    <!--
        <entire_patient_set/>
        <patient_set_coll_id>0</patient_set_coll_id>
    -->
</patient_list>
</crc:request>

```

2.6.2.6 XSI:TYPE="CRC: GETOBSERVATIONFACTBYPRIMARYKEY_REQUESTTYPE "

```

<crc:request xsi:type="crc:observation_fact_set_requestType">
    </event_id>
    </patient_id>
    </concept_cd/>
    </observer_id/>
    </start_date/>
    </modifier_cd/>
</crc:request>

```

2.6.3 Responses

2.6.3.1 XSI:TYPE="CRC:MASTER_RESPONSETYPE"

```

<crc:response xsi:type="crc:master_responseType">
    <query_master>
        <query_master_id>0</query_master_id>
        <name/>
        <user_id/>
        <group_id/>
    </query_master>
</crc:response>

```

```

        <create_date>2000-12-30T00:00:00</create_date>
        <delete_date>2000-12-30T00:00:00</delete_date>
        <request_xml/>
        <generated_sql/>
    </query_master>
    <query_master>
        <query_master_id>1</query_master_id>
        <name/>
        <user_id/>
        <group_id/>
        <create_date>2000-12-30T00:00:00</create_date>
        <delete_date>2000-12-30T00:00:00</delete_date>
        <request_xml/>
        <generated_sql/>
    </query_master>
</crc:response>

```

2.6.3.2 XSI:TYPE="CRC:INSTANCE_RESPONSETYPE"

```

<crc:response xsi:type="crc:instance_responseType">
    <query_instance>
        <query_instance_id>0</query_instance_id>
        <query_master_id>0</query_master_id>
        <user_id/>
        <group_id/>
        <batch_mode/>
        <start_date>2000-12-30T00:00:00</start_date>
        <end_date>2000-12-30T00:00:00</end_date>
        <query_status_type>
            <status_type_id>0</status_type_id>
            <name>finished</name>
            <description/>
        </query_status_type>
    </query_instance>
    <query_instance>
        <query_instance_id>1</query_instance_id>
        <query_master_id>0</query_master_id>
        <user_id/>
        <group_id/>
        <batch_mode/>
        <start_date>2000-12-30T00:00:00</start_date>
        <end_date>2000-12-30T00:00:00</end_date>
    </query_instance>
</crc:response>

```

```

    <query_status_type>
      <status_type_id>0</status_type_id>
      <name>finished</name>
      <description/>
    </query_status_type>
  </query_instance>
</crc:response>

```

2.6.3.3 XSI:TYPE="CRC:QUERY_RESULT_INSTANCETYPE"

```

<crc:response xsi:type="crc:result_responseType">
  <query_result_instance>
    <result_instance_id>0</result_instance_id>
    <query_instance_id>0</query_instance_id>
    <query_result_type>
      <result_type_id>0</result_type_id>
      <name>PATIENT_SET</name>
      <description/>
    </query_result_type>
    <set_size>0</set_size>
    <start_date>2000-12-30T00:00:00</start_date>
    <end_date>2000-12-30T00:00:00</end_date>
    <query_status_type>
      <status_type_id>0</status_type_id>
      <name>finished</name>
      <description/>
    </query_status_type>
  </query_result_instance>
  <query_result_instance>
    <result_instance_id>1</result_instance_id>
    <query_instance_id>0</query_instance_id>
    <query_result_type>
      <result_type_id>1</result_type_id>
      <name>ENCOUNTER_SET</name>
      <description/>
    </query_result_type>
    <set_size>0</set_size>
    <start_date>2000-12-30T00:00:00</start_date>
    <end_date>2000-12-30T00:00:00</end_date>
    <query_status_type>
      <status_type_id>0</status_type_id>
      <name>finished</name>

```

```

        <description/>
    </query_status_type>
</query_result_instance>
</crc:response>

```

2.6.3.4 XSI:TYPE="CRC:REQUEST_XML_RESPONSETYPE"

```

<crc:response xsi:type="crc:request_xml_responseType">
  <xml_string><![CDATA[
    <query_definition>
      <query_name/>
      <query_description/>
      <query_timing>ANY</query_timing>
      <specificity_scale>0</specificity_scale>
      <query_date_from>2000-12-30T00:00:00</query_date_from>
      <query_date_to>2000-12-30T00:00:00</query_date_to>
      <panel>
        <panel_number>0</panel_number>
        <panel_date_from>2000-12-30T00:00:00</panel_date_from>
        <panel_date_to>2000-12-30T00:00:00</panel_date_to>
        <invert>0</invert>
        <total_item_occurrences>0</total_item_occurrences>
        <item>
          <hlevel>0</hlevel>
          <item_name/>
          <item_table/>
          <item_key/>
          <item_icon/>
          <tooltip/>
          <class/>
        </item>
      </panel>
    </query_definition>
  ]]></xml_string>
</crc:response>

```

2.6.3.5 XSI:TYPE="CRC:MASTER_INSTANCE_RESULT_RESPONSETYPE"

```

<crc:response xsi:type="crc:master_instance_result_responseType">

```

```

<query_master>
  <query_master_id>0</query_master_id>
  <name/>
  <user_id/>
  <group_id/>
  <create_date>2000-12-30T00:00:00</create_date>
  <delete_date>2000-12-30T00:00:00</delete_date>
  <request_xml/>
  <generated_sql/>
</query_master>
<query_instance>
  <query_instance_id>0</query_instance_id>
  <query_master_id>0</query_master_id>
  <user_id/>
  <group_id/>
  <batch_mode/>
  <start_date>2000-12-30T00:00:00</start_date>
  <end_date>2000-12-30T00:00:00</end_date>
  <query_status_type>
    <status_type_id>0</status_type_id>
    <name>finished</name>
    <description/>
  </query_status_type>
</query_instance>
<query_result_instance>
  <result_instance_id>0</result_instance_id>
  <query_instance_id>0</query_instance_id>
  <query_result_type>
    <result_type_id>0</result_type_id>
    <name>PATIENT_SET</name>
    <description/>
  </query_result_type>
  <set_size>0</set_size>
  <start_date>2000-12-30T00:00:00</start_date>
  <end_date>2000-12-30T00:00:00</end_date>
  <query_status_type>
    <status_type_id>0</status_type_id>
    <name>finished</name>
    <description/>
  </query_status_type>
</query_result_instance>
<query_result_instance>
  <result_instance_id>1</result_instance_id>
  <query_instance_id>0</query_instance_id>
  <query_result_type>

```

```

        <result_type_id>1</result_type_id>
        <name>ENCOUNTER_SET</name>
        <description/>
    </query_result_type>
    <set_size>0</set_size>
    <start_date>2000-12-30T00:00:00</start_date>
    <end_date>2000-12-30T00:00:00</end_date>
    <query_status_type>
        <status_type_id>0</status_type_id>
        <name>finished</name>
        <description/>
    </query_status_type>
</query_result_instance>
</crc:response>

```

2.6.3.6 XSI:TYPE="CRC:INSTANCE_RESULT_RESPONSESETTYPE"

```

<crc:response xsi:type="crc:instance_result_responseType">
    <query_instance>
        <query_instance_id>0</query_instance_id>
        <query_master_id>0</query_master_id>
        <user_id/>
        <group_id/>
        <batch_mode/>
        <start_date>2000-12-30T00:00:00</start_date>
        <end_date>2000-12-30T00:00:00</end_date>
        <query_status_type>
            <status_type_id>0</status_type_id>
            <name>finished</name>
            <description/>
        </query_status_type>
    </query_instance>
    <query_result_instance>
        <result_instance_id>0</result_instance_id>
        <query_instance_id>0</query_instance_id>
        <query_result_type>
            <result_type_id>0</result_type_id>
            <name>PATIENT_SET</name>
            <description/>
        </query_result_type>
        <set_size>0</set_size>
        <start_date>2000-12-30T00:00:00</start_date>
    </query_result_instance>
</query_result_instance>
</crc:response>

```



```

    <end_date>2000-12-30T00:00:00</end_date>
    <query_status_type>
      <status_type_id>0</status_type_id>
      <name>finished</name>
      <description/>
    </query_status_type>
  </query_result_instance>
  <query_result_instance>
    <result_instance_id>1</result_instance_id>
    <query_instance_id>0</query_instance_id>
    <query_result_type>
      <result_type_id>1</result_type_id>
      <name>ENCOUNTER_SET</name>
      <description/>
    </query_result_type>
    <set_size>0</set_size>
    <start_date>2000-12-30T00:00:00</start_date>
    <end_date>2000-12-30T00:00:00</end_date>
    <query_status_type>
      <status_type_id>0</status_type_id>
      <name>finished</name>
      <description/>
    </query_status_type>
  </query_result_instance>
</crc:response>

```

2.6.3.7 XSI:TYPE="CRC:PATIENT_DATA_RESPONSETYPE"

```

<crc:response xsi:type="crc:patient_data_responseType">
  <crc:patient_data>
    <!-- see PDO cell messaging document -->
  </crc:patient_data>
</crc:response>

```

2.7 XML Schema Definitions

The CRC XML schema consists of the following XSD files:

2.7.1 CRC.xsd

This schema is not used directly to create CRC messages, but is included in other CRC_PDO_QRY.xsd and CRC_PSM_QRY.xsd.

2.7.2 CRC_PDO_QRY.xsd

This schema is used for validating CRC patient data queries and defines a <crc:header> and <crc:sql> tag.

2.7.3 CRC_PDO_QRY_request.xsd

This schema is not used directly but is included in CRC_PDO_QRY.xsd.

2.7.4 CRC_PDO_QRY_response.xsd

This schema is not used directly but is included in CRC_PDO_QRY.xsd.

2.7.5 CRC_PSM_OBJ.xsd

This schema defines the data objects that hold information about patient set queries.

2.7.6 CRC_PSM_QRY.xsd

This schema is used for validating CRC patient set queries and defines a <crc:header> and <crc:sql> tag.

2.7.7 CRC_PSM_QRY_request.xsd

This schema is not used directly but is included in CRC_PSM_QRY.xsd.

2.7.8 CRC_PSM_QRY_response.xsd

This schema is not used directly but is included in CRC_PSM_QRY.xsd.

2.7.9 CRC_PSM_QRY_query_definition.xsd

This schema validates the xml format that defines a patient set query.

2.7.10 CRC_PANEL.xsd

This schema defines the panel definition.

2.7.11 CRC_UPLOADER_QRY.xsd

This schema defines the upload request and response message.

2.7.12 I2B2_RESULT_MSG.xsd

This schema defines the result format in name-value pairs.