I2B2 on IRIS

Executive Summary

InterSystems IRIS platform is a leading technical infrastructure used in production and research settings for Healthcare and Life Sciences. The i2b2/Transmart community has implemented its query builder and underlying i2b2 core on top of three mainstream data sources - MS SQL Server, Oracle and Postgres. This POC is focused on assessment and gap analysis for adding InterSystems IRIS as an additional data source. This would allow i2b2 clients to take advantage of the IRIS high-performance data querying capabilities as well as multitude of other features and functionality offered by IRIS.

What & Why

Introduction: POC Goals

- Investigate compatibility of the i2b2 Query builder, i2b2 web and i2b2 core with IRIS backend and identify gaps preventing usage of the i2b2 Web client with IRIS back end.
- Determine steps necessary to mitigate compatibility gaps for i2b2 with IRIS backend .
- Develop a data migration path from i2b2- to IRIS.
- Execute direct SQL queries against i2b2 data in IRIS and achieve equivalence of the results with the results in the relational DB (Postgres) within limited POC scope. Capture and document the differences as well as steps to expand to a larger scope.
- Implement an infrastructure for exporting i2b2 patient data as FHIR resources, map and export sample resources (e.g. Patient, Meds) based on the data in the i2b2 instance.
- Document findings and publish to InterSystems open exchange.

How it was done

Steps creating the project

1. Set up Linux VM.
2. Install and configure InterSystems IRIS for Health 2020.3 or higher on this VM.

3. Install and configure a reference i2b2 instance on the POC Linux VM with Postgres DB as a data source with a demo patient data set
4. Configure and test i2b2 Web client (query builder) with the Postgres DB as a data source.
5. Migrate i2b2 schema and demo patient data from the reference i2b2 instance into IRIS.
6. Create and execute several representative identical queries in IRIS SQL tool and in a Postgres SQL UI (PGAdmin). Compare results and document difference between I2B2 and IRIS.
7. Create and implement I2B2 - FHIR mapping for two representative FHIR resources (e.g. Patient, MedicationRequest) and demonstrate FHIR export for a few patients.
8. Create IRIS production to export patient data from IRIS as FHIR resources (e.g. by patient_id).
9. Document the process of implementation of I2B2 on IRIS.
10. Publish the implementation and documentation to InterSystems Open Exchange.

System configuration

- system: CentOS8
- processor: AMD EPYC Processor (with IBPB)
- memory: 2000MiB System Memory
- disk: 40GB QEMU HARDDISK
- volume: 38GiB EXT4 volume
- IRIS for UNIX (Red Hat Enterprise Linux for x86-64) 2020.3 (Build 304U)
- i2b2 Core Server Source Code 1.7.12a (May, 2020)
- i2b2 Web Client 1.7.12a (May, 2020)

Data migration I2B2 >>IRIS

- The InterSystems Management Portal provides a **Server Configuration** page that allows you to install a new FHI server and then configure it. If you do not have a Foundation namespace, go to **Health**, and select **Installer Wizard** from the top menu bar. The **Configure Foundation** button allows you to create a new Foundation namespace. Be sure to activate the namespace after creating it
- Install classes and lookup tables into IRIS from the repo
- Configure FHIR framework as described in the document. Use I2B2.HS.FHIRServer.Storage.Json.InteractionsStrategy as strategy class
- Install JDBC driver
- Transfer data from i2b2 database into th IRIS by using InterSystems migrate tools as described in the document. Please note you should not `copy Definitions.`

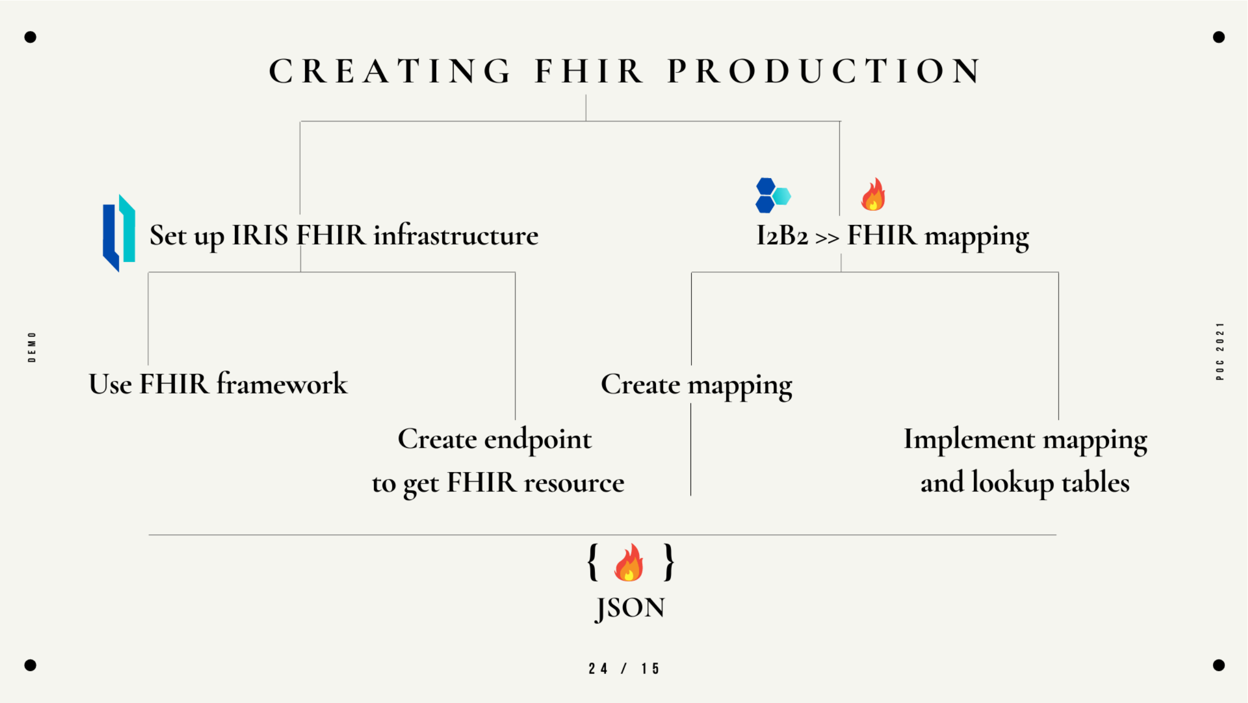
Demonstrative Queries

Create and execute several representative identical queries in IRIS SQL tool, DBeaver on IRIS DB and in a Postgres SQL UI (PGAdmin) on I2B2 data. Compare results of execution and query performance. Document changes to the SQL that is necessary for queries to work in IRIS and to return identical/similar results - IRIS optimized queries.
These queries we took from the open I2B2 community and from I2B2 test files.

Query	Description
1	Select and count symptoms reported by patients with Asthma diagnoses [ICD9:493]
2	Select other diagnoses with the same symptoms as with Asthma diagnoses.
3	Select Medications, doses and routes for Patients older than 45 years with Asthma diagnoses [ICD9:493].
4	Select Min and Max labtest results values for patients with Asthma diagnoses [ICD9:493].
5.1	Count all female patients (use patient_dimension tbl) QIQ_2Q_FEMALE_[18].xml
5.2	Count all female patients (use observation_fact tbl and concept_cd).
5.3	Count all female patients (use observation_fact tbl and concept_path from concept_dimension).
6	Count patients getting Amoxicillin (MED) dose = 500 mg MCQValNumLTE500_Dose_[17]_2000ms.xml
7	Count patients getting Medications by injection or IV route MCQValEnmMV_Route[63]_2015ms.xml
8	Count patients getting Amoxicillin (MED) every day MCQValEnmSV_Freq_[17]_2016ms.xml
9	Count patients with Diabetes mellitus Dx MQ_princDx_Diabetes_[2]_2141ms.xml
10	Count patients with Dx Chronic obstructive diseases (490-496) and with severity: FATAL, Type I Hypersensitivity between dates. MQDtBtw_[4]_3006ms.xml
11	Count patient getting Medications albuterol, fexofenadine, fluticasone nasal in special doses. MQExValNumLTandGTEandE_[7]_4389ms.xml

All queries list you can see in Appendix.

Creating FHIR production



CREATING FHIR PRODUCTION

Set up IRIS FHIR infrastructure

I2B2 >> FHIR mapping

Use FHIR framework

Create endpoint
to get FHIR resource

Create mapping

Implement mapping
and lookup tables

{ 🔥 }
JSON

DEMO

POC 2021

24 / 15

Existing mappings

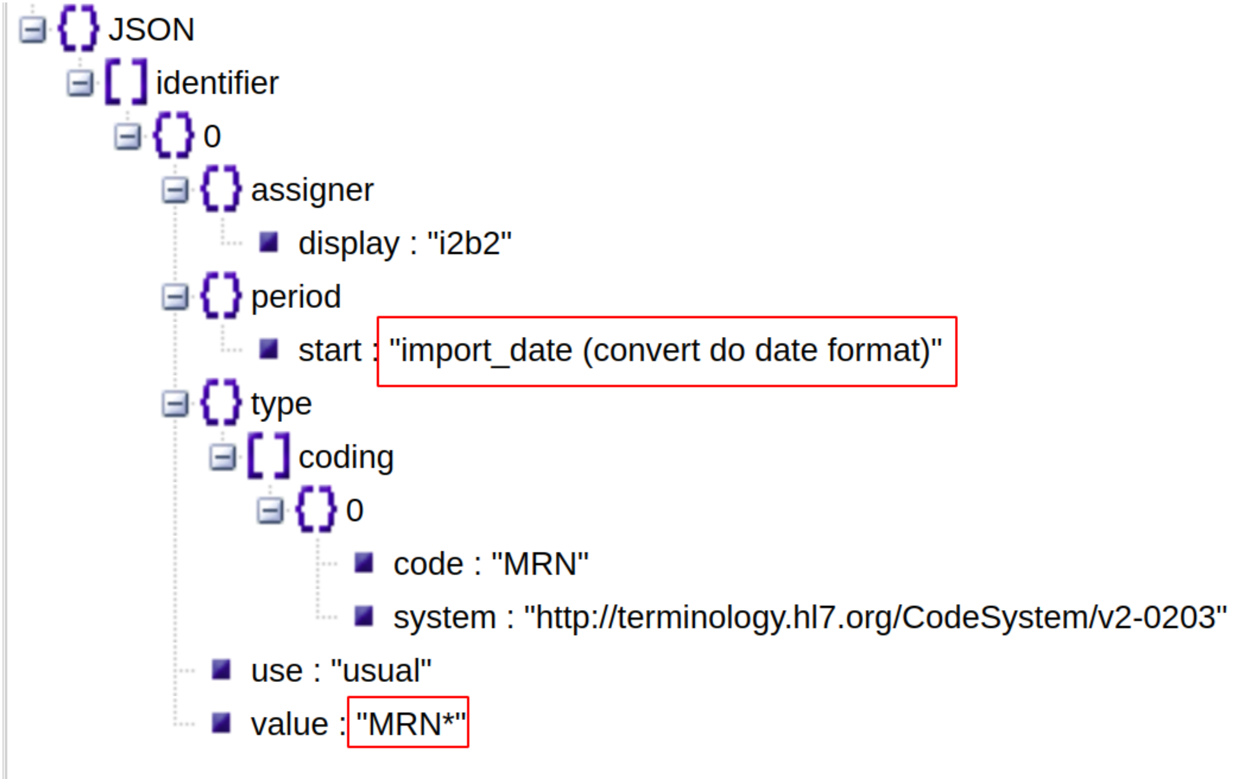
1. There is the official I2B2 > FHIR mapping was done for FHIR Release 0.1.0. It is outdated. Now we are using FHIR 4 Release 4.
2. Also there is the Python project for I2B2 >> FHIR 4 mapping. That package creates an i2b2 ontology from the FHIR STU3/R4 resource model. It uses a combination of the FHIR W5 (who, what, why, where, when) ontology and the FHIR Resource Ontology to create an i2b2 equivalent.

Patient mapping rules

There are multiple ways mapping can be done - we present one of them based on our own experience - in particular based on our extensive OMOP experience. We can tweak it as needed and expand to other resources during production implementation. The mapping is independent and will not affect the FHIR export infrastructure that we put together.

Identifier

*select **import_date**, tval_char as **MRN** from public.observation_fact where concept_cd like 'PHI:MRN' and patient_num = patient_num

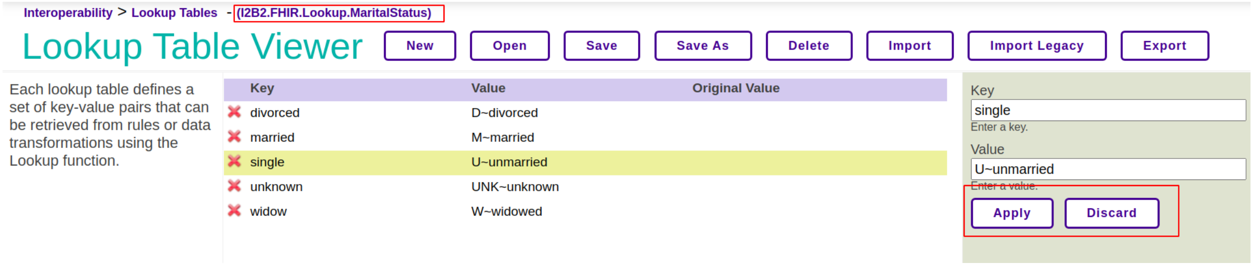


```
{} JSON
  [] identifier
    {} 0
      {} assigner
        display : "i2b2"
      {} period
        start : "import_date (convert do date format)"
      {} type
        [] coding
          {} 0
            code : "MRN"
            system : "http://terminology.hl7.org/CodeSystem/v2-0203"
      use : "usual"
      value : "MRN*"
```

Please see the Patient Mapping in Appendix.

Look up tables

To manage mapping values (edit existing ones or add new values) we use IRIS lookup tables. Path to them: *Home > Interoperability > Configure > Data Lookup Tables > Go > Open* Then you need to choose the table e.g. I2B2.FHIR.Lookup.MaritalStatus
Column Key is the value from I2B2 data base and Value column is the data we put in the FHIR resource.



Interoperability > Lookup Tables - (I2B2.FHIR.Lookup.MaritalStatus)

Lookup Table Viewer

Each lookup table defines a set of key-value pairs that can be retrieved from rules or data transformations using the Lookup function.

Key	Value	Original Value
divorced	D~divorced	
married	M~married	
single	U~unmarried	
unknown	UNK~unknown	
widow	W~widowed	

Key: single
Enter a key.
Value: U~unmarried
Enter a value.

Patient resource request

To implement FHIR mapping we used the FHIR Server. The Service is a singleton class that allows only one instance of itself to be instantiated for an endpoint. This instantiation occurs when the first FHIR request is sent to the Service by the REST Handler or Business Operation; once instantiated, the Service exists until the process ends. For server applications making

FHIR requests programmatically, the app must call **HS.FHIRServer.Service.EnsureInstance()** to retrieve the Service before sending the first request.

In most cases, the Service class (**HS.FHIRServer.Service**) is ready to uphold the endpoint's FHIR standard and route requests without being subclassed. Custom logic that determines how the FHIR server behaves is written into the Interactions and InteractionsStrategy subclasses, not the Service.

URL: http://IP:PORT/i2b2/fhir/r4/Patient/PATIENT_ID
METOD: GET
HEADER:
KEY = Accept
VALUE = application/json
AUTHORIZATION:
TYPE: Basic auth
LOGIN:
PASSWORD:

You should get JSON as a result.

Results

Queries speed evaluation

In I2B2 column is the time that we got running a query on Postgres using PGAdmin tool. Iris column is the time we get performing exactly the same query in IRIS SQL tool. Iris optimized is the result we got after we add some indices or change the syntax.
Some queries were good enough that we ever didn't need to do any optimization (queries 5,6,7,11).

Query	Results equal	I2b2	Iris	IRIS optimized
1	+	691 msec	1321 msec	430 msec
2	+	2 secs 94 msec	30000	417 msec
3	+	747 msec	1003 msec	160 msec
4	+	611 msec	6757 msec	??
5.1	+	271 msec	53 msec	
5.2	+	260 msec	3 msec	

5.3	+	493 msec	4 msec	
6	+	619 msec	9 msec	
7	+	280 msec	5 msec	
8	+	347 msec	826 msec	11 msec
9	+	316 msec	238 msec	29 msec
10	+	572 msec	816 msec	32 msec
11	+	513 msec	15 msec	

Syntax difference

In IRIS SQL tool the following changes should be done:

- In GROUP BY clause you need to use field name not alias from select (e.g. group by c1.name_char, not group by second_dx).
- Instead of LIKE you need to use Contains operator ([) to deal with case sensitive search.
- Instead of LIKE operator you need to use %STARTSWITH (e.g. concept_cd %STARTSWITH 'ICD9:493', but not LIKE 'ICD9:493%').
- Regex function name in IRIS is %MATCHES

Patient FHIR resource output

This is the JSON that we get after the http request to IRIS Production.

```
{"resourceType":"Patient","address":[{"city":"Cambridge","country":"US","postalCode":"02140","state":"Massachusetts","type":"both","use":"home"}],"birthDate":"1976-11-23","communication":[{"language":{"coding":[{"code":"en","display":"english","system":"http://hl7.org/fhir/ValueSet/languages"}]},"preferred":true}],"deceasedBoolean":false,"extension":[{"url":"http://hl7.org/fhir/StructureDefinition/patient-nationality","valueCoding":{"code":"2135-2","display":"Hispanic or Latino","system":"http://terminology.hl7.org/CodeSystem/v3-Ethnicity"}},{"url":"http://hl7.org/fhir/StructureDefinition/patient-religion","valueCoding":{"code":"1007","display":"Atheism","system":"http://terminology.hl7.org/CodeSystem/v3-ReligiousAffiliation"}}],"gender":"female","id":"1000000010","identifier":[{"assigner":{"display":"i2b2"},"period":{"start":"2010-11-04"},"type":{"coding":[{"code":"PLAC","system":"http://terminology.hl7.org/CodeSystem/v2-0203"}]},"use":"usual","value":"1000000010"},{"assigner":{"display":"i2b2"},"period":{"start":"2007-03-20"},"type":{"coding":[{"code":"MRN","system":"http://terminology.hl7.org/CodeSystem/v2-0203"}]},"use":"usual","value":"FAKEMRN:362247348"}],"maritalStatus":{"coding":[{"code":"M","display":"married","system":"http://terminology.hl7.org/CodeSystem/v3-MaritalStatus"}]},"name":[{"family":"Miller","given":["Barbara"],"use":"usual"}]}
```

Conclusion and Next Steps

1. Configure I2B2 CORE to use IRIS DB as its operational base.

I2B2 always checks the driver it is working with. But check is done by driver name comparison, and only for three supported databases: MS SQL, Postgres, Oracle. Other databases are not supported because of this.
For example, this is the error we encountered during PoC.
First it is performed SQL query: sql = "select * from pm_global_params where status_cd = 'A' and param_name_cd ='PM_LOCKED_WAIT_TIME'";
Then it checks that DB is Oracle or MS or Postgres. There is no exception in this place so we can not use any other DB type. So the next query performs incorrectly: int results = jt.queryForObject(sql, Integer.class, userId); There are a lot of places where similar coding practice is used, all of them have to be updated.
Hack:
IRIS driver has MS compatibility mode, it was modified to pretend it is Microsoft sql server. It allows us to connect I2B2 CORE to IRIS DB but not all features work correctly. The issues are with key words (as "public"), date/time requests. Each issue needs to be analyzed and processed correctly.

2. Use IRIS NLP to add text index to concept data to speed up the search.
3. Map all I2B2 data into FHIR resources.
4. Create predefined queries to speed up the analytics.

Links

- Install and configure i2b2 - https://www.i2b2.org/software/projects/hivecore/i2b2QuickInstall.pdf
- Install classes and lookup tables into IRIS from the repo - https://github.com/ELynx/i2b2-on-iris/tree/master/src
- Configure FHIR framework as described in the document - https://docs.intersystems.com/irisforhealthlatest/csp/docbook/Doc.View.cls?KEY=HXFHIR.
- Transfer data from i2b2 database into th IRIS - https://docs.intersystems.com/irisforhealthlatest/csp/docbook/DocBook.UI.Page.cls?KEY=BSQG_odbc#BSQG_odbc_migration.
- I2B2 test files - https://github.com/i2b2/i2b2-core-server/tree/master/edu.harvard.i2b2.crc/testfiles
- The official I2B2 > FHIR mapping - http://build.fhir.org/ig/HL7/cdmh/profiles.html
- The Python project for I2B2 >> FHIR 4 mapping - https://github.com/BD2KOnFHIR/i2FHIRb2

- The FHIR Server - https://docs.intersystems.com/healthconnectlatest/csp/docbook/Doc.View.cls?KEY=HXFHIR_server_intro#HXFHIR_server_arch
- JDBC driver configuration error - https://github.com/i2b2/i2b2-core-server/blob/6f83a8f1ff4d2f5b3e55972792cfb07d9446785a/edu.harvard.i2b2.pm/src/edu/harvard/i2b2/pm/dao/PMDbDao.java#L1110

Appendix

Script example to create I2B2 table structure

This is the sample script to create the I2B2 table Observation_fact.

```
CREATE TABLE "public"."observation_fact"(
"encounter_num" INTEGER NOT NULL,
"patient_num" INTEGER NOT NULL,
"concept_cd" VARCHAR(50) NOT NULL,
"provider_id" VARCHAR(50) NOT NULL,
"start_date" TIMESTAMP NOT NULL,
"modifier_cd" VARCHAR(100) NOT NULL DEFAULT '@',
"instance_num" INTEGER NOT NULL DEFAULT 1,
"valtype_cd" VARCHAR(50),
"tval_char" VARCHAR(255),
"nval_num" NUMERIC(18,5),
"valueflag_cd" VARCHAR(50),
"quantity_num" NUMERIC(18,5),
"units_cd" VARCHAR(50),
"end_date" TIMESTAMP,
"location_cd" VARCHAR(50),
"observation_blob" VARCHAR(2147483647),
"confidence_num" NUMERIC(18,5),
"update_date" TIMESTAMP,
"download_date" TIMESTAMP,
"import_date" TIMESTAMP,
"sourcesystem_cd" VARCHAR(50),
"upload_id" INTEGER,
"text_search_index" INTEGER,
CONSTRAINT "observation_fact_pk" PRIMARY KEY( "patient_num", "concept_cd",
"modifier_cd", "start_date", "encounter_num", "instance_num", "provider_id")
)
```

Queries description

Query 1

Select and count symptoms reported by patients with Asthma diagnoses [ICD9:493]

Postgres	select count (patient_num) as N, c.name_char as symp from observation_fact as f left join concept_dimension as c on c.concept_cd = f.concept_cd where patient_num in (SELECT distinct patient_num FROM observation_fact where concept_cd like 'ICD9:493%') and c.concept_path like '%Symptoms%' group by symp order by N desc
IRIS	select count (patient_num) as N, c.name_char as symp from observation_fact as f left join concept_dimension as c on c.concept_cd = f.concept_cd where patient_num in (SELECT distinct patient_num FROM observation_fact where concept_cd %STARTSWITH 'ICD9:493') and c.concept_path like 'Symptoms' group by c.name_char order by N desc
IRIS optimized	select count(f.patient_num) as N, c.name_char as symp from observation_fact as f left join concept_dimension as c on c.concept_cd = f.concept_cd left join observation_fact of2 on of2.patient_num = f.patient_num where of2.concept_cd %STARTSWITH 'ICD9:493' and c.concept_path **[** 'Symptoms' group by c.name_char order by N desc

Query 2

Select other diagnoses with the same symptoms as with Asthma diagnoses

Postgres	Select count (distinct patient_num) as num, c1.name_char as second_dx, c1.concept_cd ,T.symp from observation_fact as f1 left join concept_dimension as c1 on c1.concept_cd = f1.concept_cd left join (select distinct patient_num as N, c.name_char as symp

	from observation_fact as f left join concept_dimension as c on c.concept_cd = f.concept_cd where patient_num in (SELECT distinct patient_num FROM observation_fact where concept_cd like 'ICD9:493%') and c.concept_path like '%Symptoms%') as T on T.N = f1.patient_num where c1.concept_path like '%Diagnoses%' and c1.concept_path not like '%Symptoms%' and c1.concept_cd not like 'ICD9:493%' and c1.concept_cd SIMILAR TO 'ICD9:[0-9]*' group by second_dx, c1.concept_cd,T.symp order by second_dx asc ,num desc
IRIS	Select count (distinct patient_num) as num, c1.name_char, c1.concept_cd ,T.symp from observation_fact as f1 left join concept_dimension as c1 on c1.concept_cd = f1.concept_cd left join (select distinct patient_num as N, c.name_char as symp from observation_fact as f left join concept_dimension as c on c.concept_cd = f.concept_cd where f.patient_num in (SELECT distinct patient_num FROM observation_fact where concept_cd %STARTSWITH 'ICD9:493') and c.concept_path ['Symptoms') as T on T.N = f1.patient_num where c1.concept_path like '%Diagnoses%' and c1.concept_path not like '%Symptoms%' and c1.concept_cd not like 'ICD9:493%' and c1.concept_cd %MATCHES 'ICD9:[0-9]*' group by c1.name_char, c1.concept_cd,T.symp order by c1.name_char asc ,num desc

IRIS optimized	Select count (distinct patient_num) as num, c1.name_char, c1.concept_cd ,T.symp
	from observation_fact as f1
	join concept_dimension as c1
	    on c1.concept_cd = f1.concept_cd
	join (select distinct patient_num as N, c.name_char as symp from observation_fact as f
	        left join concept_dimension as c
	            on c.concept_cd = f.concept_cd
	        where f.patient_num in (SELECT distinct patient_num FROM observation_fact where concept_cd %STARTSWITH 'ICD9:493')

	and c.concept_path ['Symptoms') as T on T.N = f1.patient_num where c1.concept_cd %STARTSWITH 'ICD9:' and (not c1.concept_cd %STARTSWITH 'ICD9:493') and c1.concept_cd %PATTERN '1"ICD9:"1.N' and c1.concept_path ['Diagnoses' and (not c1.concept_path ['Symptoms') group by c1.name_char, c1.concept_cd, T.symp order by c1.name_char asc, num desc

Don’t need “LEFT” join here. It causes extra rows useless for GROUP clause.

QUERY 3

Select Medications, doses and routes for Patients older than 45 years with Asthma diagnoses [ICD9:493]

Postgres	select count(f.encounter_num) as N, c.name_char as Med, f.nval_num as dose , f.units_cd as units, T.route from observation_fact as f left join concept_dimension as c on c.concept_cd = f.concept_cd join (select f1.encounter_num, f1.instance_num, f1.tval_char as route from observation_fact as f1 left join concept_dimension as c1 on c1.concept_cd = f1.concept_cd where patient_num in (SELECT distinct f2.patient_num FROM observation_fact as f2 join patient_dimension as d on d.patient_num = f2.patient_num where concept_cd like 'ICD9:493%' and d.age_in_years_num >= 45) and c1.concept_path like '%Medications%' and modifier_cd = 'MED:ROUTE') as T on T.encounter_num = f.encounter_num and T.instance_num = f.instance_num where f.modifier_cd = 'MED:DOSE' group by dose, units, Med, T.route

	order by Med asc
IRIS	select count(f.encounter_num) as N, c.name_char as Med, f.nval_num as dose , f.units_cd as units, T.route from observation_fact as f left join concept_dimension as c on c.concept_cd = f.concept_cd join (select f1.encounter_num, f1.instance_num, f1.tval_char as route from observation_fact as f1 left join concept_dimension as c1 on c1.concept_cd = f1.concept_cd where patient_num in (SELECT distinct f2.patient_num FROM observation_fact as f2 join patient_dimension as d on d.patient_num = f2.patient_num where concept_cd %STARTSWITH 'ICD9:493' and d.age_in_years_num >= 45) and c1.concept_path like '%Medications%' and modifier_cd = 'MED:ROUTE') as T on T.encounter_num = f.encounter_num and T.instance_num = f.instance_num where f.modifier_cd = 'MED:DOSE' group by f.nval_num, f.units_cd, c.name_char, T.route order by Med asc
IRIS optimized	select count(f.encounter_num) as N, c.name_char as Med, f.nval_num as dose , f.units_cd as units, T.route from observation_fact as f left join concept_dimension as c on c.concept_cd = f.concept_cd join (select f1.encounter_num, f1.instance_num, f1.tval_char as route from observation_fact as f1 left join concept_dimension as c1 on c1.concept_cd = f1.concept_cd where patient_num in (SELECT distinct f2.patient_num FROM observation_fact as f2

	join patient_dimension as d on d.patient_num = f2.patient_num where concept_cd %STARTSWITH 'ICD9:493' and d.age_in_years_num >= 45) and c1.concept_path **[** 'Medications' and modifier_cd = 'MED:ROUTE') as T on T.encounter_num = f.encounter_num and T.instance_num = f.instance_num where f.modifier_cd = 'MED:DOSE' group by f.nval_num, f.units_cd, c.name_char, T.route order by Med asc

QUERY 4

Select Min and Max labtest results values for patients with Asthma diagnoses [ICD9:493]

Postgres	select count (patient_num) as N, lower(c.name_char) as tests, min(nval_num), max(nval_num),units_cd as units from observation_fact as f left join concept_dimension as c on c.concept_cd = f.concept_cd where patient_num in (SELECT distinct patient_num FROM observation_fact where concept_cd like 'ICD9:493%') and c.concept_path like '%Labtests%' and nval_num is not NULL group by tests, units_cd order by N desc
IRIS	select count (patient_num) as N, c.name_char as tests, min(nval_num), max(nval_num),units_cd as units from observation_fact as f left join concept_dimension as c on c.concept_cd = f.concept_cd where patient_num in (SELECT distinct patient_num FROM observation_fact where concept_cd %STARTSWITH 'ICD9:493') and c.concept_path ['Labtests' and nval_num is not NULL group by c.name_char, units_cd order by tests, N desc
IRIS optimized	Not needed

QUERY 5.1

Count all female patients (use patient_dimension tbl) QIQ_2Q_FEMALE_[18].xml

Postgres	select count (patient_num) as N from public.patient_dimension where sex_cd = 'F'
IRIS	select count (patient_num) as N from public.patient_dimension where sex_cd = 'F'
IRIS optimized	Not needed

QUERY 5.2

Count all female patients (use observation_fact tbl and concept_cd)

Postgres	select count(distinct patient_num) from observation_fact where concept_cd = 'DEM|SEX:f'
IRIS	select count(distinct patient_num) from observation_fact where concept_cd = 'DEM|SEX:f'
IRIS optimized	Not needed

QUERY 5.3

Count all female patients (use observation_fact tbl and concept_path from concept_dimension)

Postgres	select count(distinct patient_num) from observation_fact as o join concept_dimension as c on c.concept_cd = o.concept_cd where c.name_char = 'Female' and concept_path = '\i2b2\Demographics\Gender\Female\'
IRIS	select count(distinct patient_num) from observation_fact as o join concept_dimension as c on c.concept_cd = o.concept_cd where c.name_char = 'Female' and concept_path = '\i2b2\Demographics\Gender\Female\'
IRIS optimized	Not needed

QUERY 6

Count patients getting Amoxicillin (MED) dose = 500 mg
MCQValNumLTE500_Dose_[17]_2000ms.xml

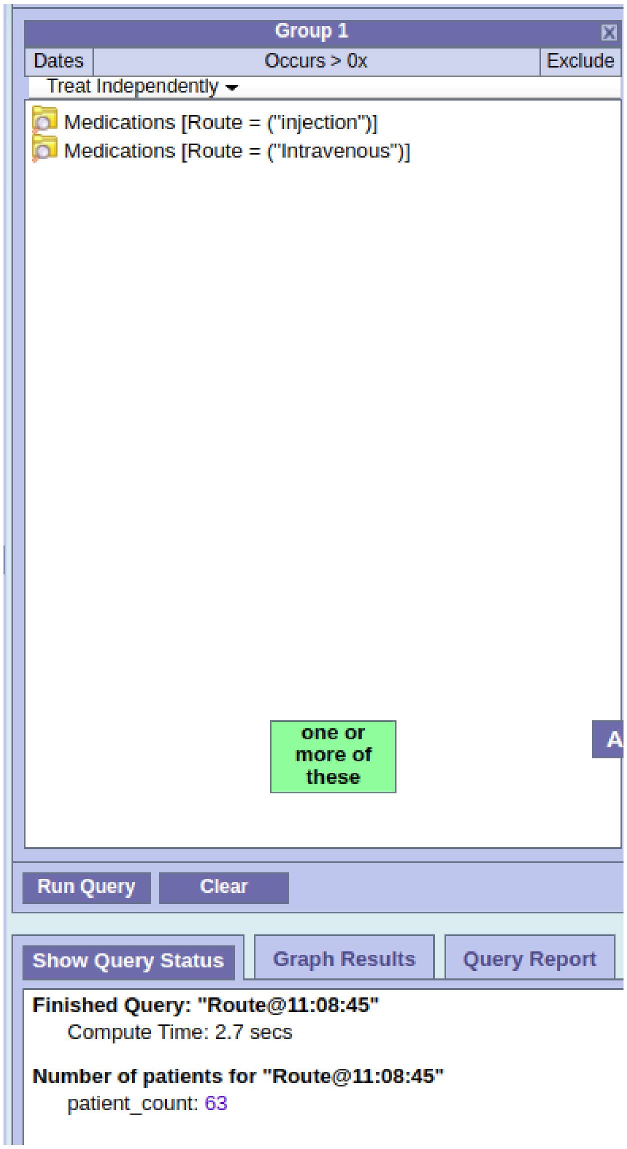
Postgres	select count(distinct patient_num) from observation_fact as o join concept_dimension as c on c.concept_cd = o.concept_cd where concept_path like '%amoxicillin%' and modifier_cd = 'MED:DOSE' and units_cd = 'mg' and nval_num = '500'
IRIS	select count(distinct patient_num) from observation_fact as o join concept_dimension as c on c.concept_cd = o.concept_cd where concept_path like '%amoxicillin%' and modifier_cd = 'MED:DOSE' and units_cd = 'mg' and nval_num = '500'
IRIS optimized	Not needed

QUERY 7

Count patients getting Medications by injection or IV route
MCQValEnmMV_Route[63]_2015ms.xml

Postgres	select count(distinct patient_num) from observation_fact where modifier_cd = 'MED:ROUTE' and tval_char in ('IV', 'injection')
IRIS	select count(distinct patient_num) from observation_fact where modifier_cd = 'MED:ROUTE' and tval_char in ('IV', 'injection')
IRIS optimized	Not needed

Web client



Medications [Route = ("injection")]
Medications [Route = ("Intravenous")]

Finished Query: "Route@11:08:45"
Compute Time: 2.7 secs

Number of patients for "Route@11:08:45"
patient_count: 63

QUERY 8

Count patients getting Amoxicillin (MED) every day
MCQValEnmSV_Freq_[17]_2016ms.xml

Postgres	select count(distinct patient_num) from observation_fact as o join concept_dimension as c on c.concept_cd = o.concept_cd where concept_path like '%amoxicillin%' and modifier_cd = 'MED:FREQ' and tval_char = 'QD'

IRIS	select count(distinct patient_num) from observation_fact as o join concept_dimension as c on c.concept_cd = o.concept_cd where concept_path like '%amoxicillin%' and modifier_cd = 'MED:FREQ' and tval_char = 'QD'
IRIS optimized	select count(distinct patient_num) from observation_fact as o join concept_dimension as c on c.concept_cd = o.concept_cd where concept_path ['amoxicillin' and modifier_cd = 'MED:FREQ' and tval_char = 'QD'

QUERY 9

Count patients with Diabetes mellitus Dx
MQ_princDx_Diabetes_[2]_2141ms.xml

Postgres	select count(distinct patient_num) from observation_fact as o join modifier_dimension as m on o.modifier_cd = m.modifier_cd join public.concept_dimension as c on c.concept_cd = o.concept_cd where m.modifier_path = '\Principal Diagnosis\' and c.name_char like 'Diabetes mellitus%'
IRIS	select count(distinct patient_num) from observation_fact as o join modifier_dimension as m on o.modifier_cd = m.modifier_cd join concept_dimension as c on c.concept_cd = o.concept_cd where m.modifier_path = '\Principal Diagnosis\' and c.name_char %STARTSWITH 'Diabetes mellitus'
IRIS optimized	Add index for column name_char

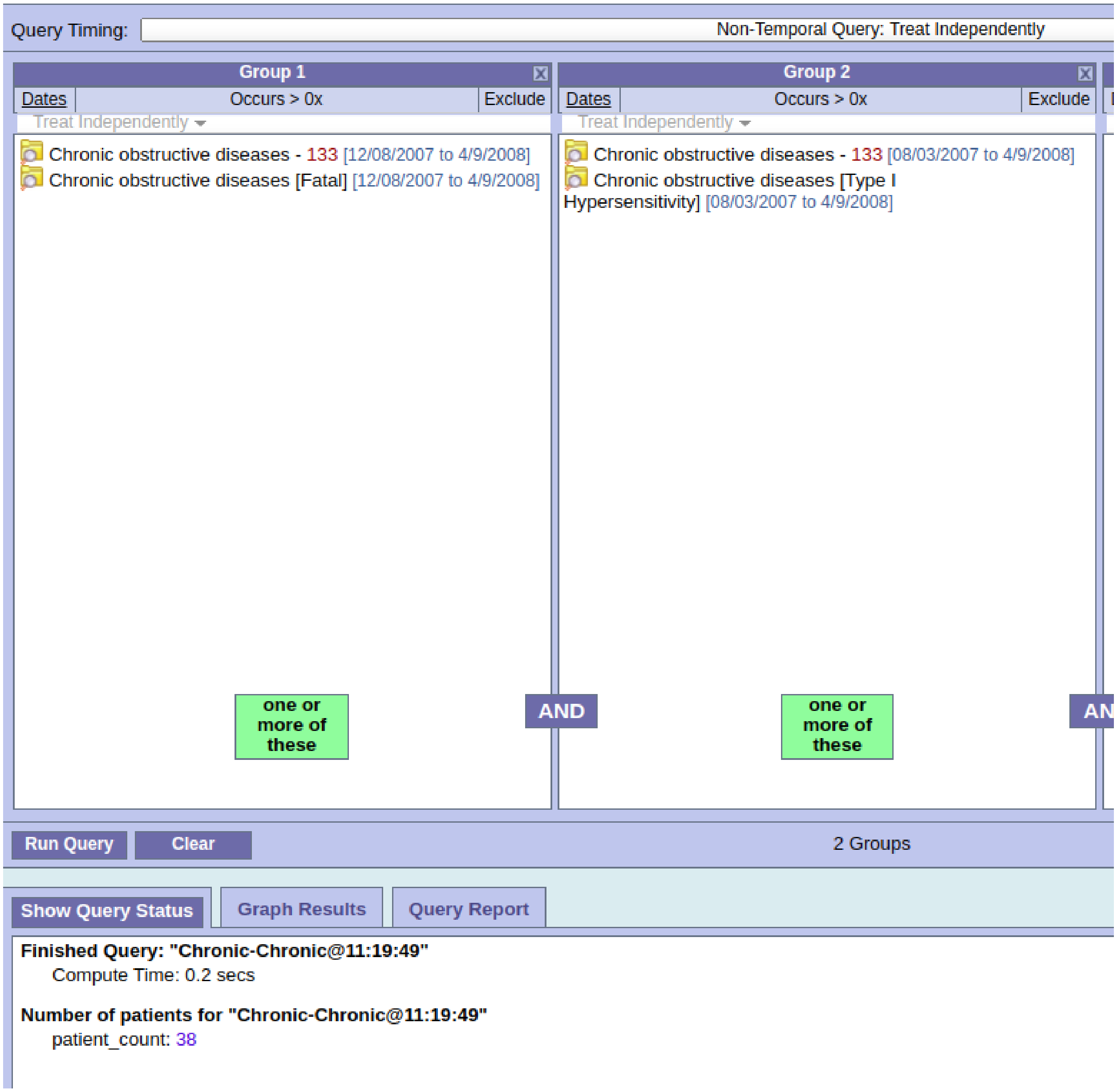
QUERY 10

Count patients with Dx Chronic obstructive diseases (490-496) and with severity: FATAL, Type I Hypersensitivity between dates
MQDtBtw_[4]_3006ms.xml

Postgres	select count(patient_num) as N, m.name_char as Severity from observation_fact as o

	join concept_dimension as c on c.concept_cd = o.concept_cd join modifier_dimension as m on m.modifier_cd = o.modifier_cd where o.modifier_cd in ('SNO:399166001', 'SNO:264774001') and concept_path like '%Chronic obstructive diseases (490-496)%' and start_date >= '2007-03-08 00:00:00' and end_date <= '2008-09-04 00:00:00' group by m.name_char
IRIS	select count(patient_num) as N, m.name_char as Severity from observation_fact as o join concept_dimension as c on c.concept_cd = o.concept_cd join modifier_dimension as m on m.modifier_cd = o.modifier_cd where o.modifier_cd in ('SNO:399166001', 'SNO:264774001') and concept_path like '%Chronic obstructive diseases (490-496)%' and start_date >= '2007-03-08 00:00:00' and end_date <= '2008-09-04 00:00:00' group by m.name_char
IRIS optimized	select count(patient_num) as N, m.name_char as Severity from observation_fact as o join concept_dimension as c on c.concept_cd = o.concept_cd join modifier_dimension as m on m.modifier_cd = o.modifier_cd where o.modifier_cd in ('SNO:399166001', 'SNO:264774001') and concept_path ['Chronic obstructive diseases (490-496)' and start_date >= '2007-03-08 00:00:00' and end_date <= '2008-09-04 00:00:00' group by m.name_char

Web client



Group 1:
Chronic obstructive diseases - 133 [12/08/2007 to 4/9/2008]
Chronic obstructive diseases [Fatal] [12/08/2007 to 4/9/2008]

Group 2:
Chronic obstructive diseases - 133 [08/03/2007 to 4/9/2008]
Chronic obstructive diseases [Type I Hypersensitivity] [08/03/2007 to 4/9/2008]

Finished Query: "Chronic-Chronic@11:19:49"
Compute Time: 0.2 secs

Number of patients for "Chronic-Chronic@11:19:49"
patient_count: 38

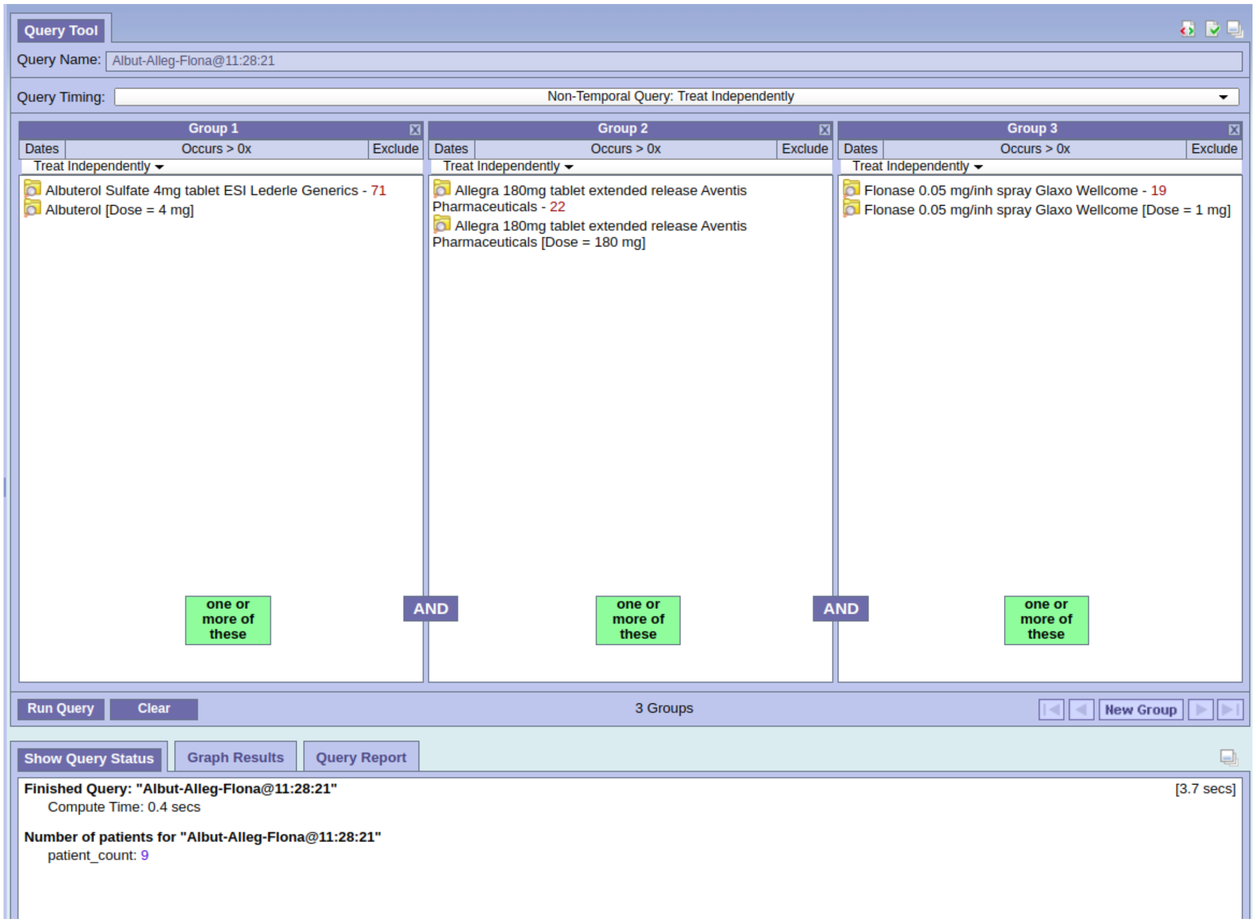
QUERY 11

Count patient getting Medications albuterol, fexofenadine, fluticasone nasal in special doses
MQExValNumLTandGTEandE_[7]_4389ms.xml

Postgres	select count(distinct patient_num), c.name_char from observation_fact as o join concept_dimension as c on c.concept_cd = o.concept_cd where concept_path like '%albuterol\\00005306343%' and modifier_cd = 'MED:DOSE' and units_cd = 'mg' and nval_num = '4' group by c.name_char

	union all select count(distinct patient_num), c.name_char from observation_fact as o join concept_dimension as c on c.concept_cd = o.concept_cd where concept_path like '%fexofenadine\\00088110947%' and modifier_cd = 'MED:DOSE' and units_cd = 'mg' and nval_num = '180' group by c.name_char union all select count(distinct patient_num), c.name_char from observation_fact as o join concept_dimension as c on c.concept_cd = o.concept_cd where concept_path like '%fluticasone nasal\\00173045301%' and modifier_cd = 'MED:DOSE' and units_cd = 'mg' and nval_num = '0.1' group by c.name_char
IRIS	select count(distinct patient_num), c.name_char from observation_fact as o join concept_dimension as c on c.concept_cd = o.concept_cd where concept_path like '%albuterol\00005306343%' and modifier_cd = 'MED:DOSE' and units_cd = 'mg' and nval_num = '4' group by c.name_char union all select count(distinct patient_num), c.name_char from observation_fact as o join concept_dimension as c on c.concept_cd = o.concept_cd where concept_path like '%fexofenadine\00088110947%' and modifier_cd = 'MED:DOSE' and units_cd = 'mg' and nval_num = '180' group by c.name_char union all select count(distinct patient_num), c.name_char from observation_fact as o join concept_dimension as c on c.concept_cd = o.concept_cd where concept_path like '%fluticasone nasal\00173045301%' and modifier_cd = 'MED:DOSE' and units_cd = 'mg' and nval_num = '0.1' group by c.name_char
IRIS optimized	Not needed

Web client



Query Name: Albut-Alleg-Flona@11:28:21

Query Timing: Non-Temporal Query: Treat Independently

Group 1
- Albuterol Sulfate 4mg tablet ESI Lederle Generics - 71
- Albuterol [Dose = 4 mg]

Group 2
- Allegra 180mg tablet extended release Aventis Pharmaceuticals - 22
- Allegra 180mg tablet extended release Aventis Pharmaceuticals [Dose = 180 mg]

Group 3
- Flonase 0.05 mg/inh spray Glaxo Wellcome - 19
- Flonase 0.05 mg/inh spray Glaxo Wellcome [Dose = 1 mg]

3 Groups

Finished Query: "Albut-Alleg-Flona@11:28:21" [3.7 secs]
Compute Time: 0.4 secs

Number of patients for "Albut-Alleg-Flona@11:28:21"
patient_count: 9

FHIR Resources - Patient Mapping

This is our custom FHIR mapping for resource Patient.

Table : public.patient_dimension, observation_fact

i2b2	**fhir element**	**fhir resource**	**rule [HARDCODE]**

patient_num,
import_date
	Patient.
	identifier
		Patient	"identifier": [
			{
			"use": "usual",
			"type": {
			"coding": [
			{
			"system":
			"http://terminology.hl7.org/CodeSystem/v2-0203",
			"code": "PLAC"
			}]},
			"value": **patient_num**

			"period": {
			"start": **import_date (*convert do date format)***
			},
			"assigner": {
			"display": "i2b2"}},
			{
			"use": "usual",
			"type": {
			"coding": [
			{
			"system":
			"http://terminology.hl7.org/CodeSystem/v2-0203",
			"code": "MR"
			}]},
			"value": **MRN***
			"period": {
			"start": **import_date (*convert do date format)***
			},
			"assigner": {
			"display": "i2b2" }}
			]
			** select tval_char as **MRN** from public.observation_fact where concept_cd like 'PHI:MRN' and patient_num = **patient_num***

birth_date	Patient.birthDate	Patient	convert from datetime into date formate

public.observation_fact table	Patient.name	Patient	"name": [
			{
			"use": "official",
			"family": **LNAME***
			"given": **FNAME****
			}
			* select tval_char as **LNAME** from public.observation_fact where concept_cd like 'PHI:LNAME' and patient_num = **patient_num**
			** select tval_char as **FNAME** from public.observation_fact where concept_cd like 'PHI:FNAME' and patient_num = **patient_num**

Death_date	Patient.deceased. deceasedBoolean
	Patient.deceased.deceasedDateTime
		Patient	if i2b2.Death_date not Null:
			Patient.deceased. deceasedBoolean = TRUE
			Patient.deceased.deceasedDateTime = Death_date
			Else:
			deceasedBoolean = false

sex_cd	Patient.gender	Patient	*select sex_cd as **GENDER** from public.patient_dimension where patient_num = patient_num

			case **GENDER** = 'F' then Patient.gender = 'Female'
			case **GENDER** = 'M' then Patient.gender = 'Male'
			case **GENDER** is NULL then Patient.gender = 'Unknown'

Race_cd	Patient.extension: patient-nationality	patient-nationality	" ""extension"": [{
			""url"":
			"http://hl7.org/fhir/StructureDefinition/patient-nationality"
			""valueCoding"":
			{
			""system"":
			"http://terminology.hl7.org/CodeSystem/v3-Ethnicity",
			""code"": CODE*,
			""display"": DISPLAY*
			} }]
			* case Race_cd != ""hispanic"" then CODE = '2186-5' and DISPLAY = 'Not Hispanic or Latino'
			case Race_cd = ""hispanic"" then CODE = '2135-2' and DISPLAY = 'Hispanic or Latino'
			"

Race_cd	Patient.extension: us-core-race	us-core-patient	"extension": [{
			"url":
			"http://hl7.org/fhir/us/core/StructureDefinition/us-core-race"
			"valueCoding":
			{
			"system":
			"http://terminology.hl7.org/CodeSystem/v3-Race",
			"code": **CODE***,
			"display": **DISPLAY***
			} }]
			* case **Race_cd** = "indian" then **CODE** = ' 1002-5' and **DISPLAY** = 'American Indian or Alaska Native**'**
			case **Race_cd** = "asian" then **CODE** = ' 2028-9' and **DISPLAY** = **'**asian'
			case **Race_cd** = **"**black" then **CODE** = **'**2054-5' and **DISPLAY** = 'Black or African American'
			case **Race_cd** = "white" then **CODE** = '2106-3' and **DISPLAY** = **'**White'
			case **Race_cd** = "hispanic" then **CODE** = '2131-1' and **DISPLAY** = 'Other Race'

religion_cd	Patient.extension: patient-religion	patient-religion	"extension": [{
			"url":
			"http://hl7.org/fhir/StructureDefinition/patient-religion"
			"valueCoding":
			{

			""system"":
			"http://terminology.hl7.org/CodeSystem/v3-ReligiousAffiliation",
			""code"": **CODE***,
			""display"": **DISPLAY***
			}
			* case **religion_cd** = ""atheist"" then **CODE** = ' 1007' and **DISPLAY** = 'Atheism'
			case **religion_cd** = ""christian"" then **CODE** = ' 1013' and **DISPLAY** = 'Christian (non-Catholic, non-specific)'
			case **religion_cd** = ""roman catholic"" then **CODE** = '1041' and **DISPLAY** = 'Roman Catholic Church'
			case **religion_cd** = ""jewish"" then **CODE** = '1026' and **DISPLAY** = 'Judaism'
			case **religion_cd** = ""muslim"" then **CODE** = '1023' and **DISPLAY** = 'Islam'
			case **religion_cd** = ""agnostic"" then **CODE** = '1004' and **DISPLAY** = 'Agnosticism'
			"

language_cd	Patient.communication. language
	Patient.communication. preferred
		Patient	Patient.communication. preferred = TRUE
			Patient.communication. language
			{
			"system": "http://hl7.org/fhir/ValueSet/languages",
			"code": **CODE***,
			"display": **DISPLAY***
			}
			* case language_cd = "english" then **CODE** = 'en' and **DISPLAY** = 'english'
			case language_cd = "german" then **CODE** = 'de' and **DISPLAY** = 'german'
			case language_cd = "spanish" then **CODE** = 'es' and **DISPLAY** = 'spanish'
			case language_cd is NULL then skip block

marital_status_cd	Patient.maritalStatus	Patient	{
			"system":
			"http://terminology.hl7.org/CodeSystem/v3-MaritalStatus",
			"code": **CODE***,
			"display": **DISPLAY***
			}
			* case marital_status_cd = "single" then **CODE** = 'U' and **DISPLAY** = 'unmarried'
			case marital_status_cd = "divorced" then **CODE** = 'D' and **DISPLAY** = 'divorced'
			case marital_status_cd = "widow" then **CODE** = 'W' and **DISPLAY** = 'widowed'
			case marital_status_cd = "married" then **CODE** = 'M' and **DISPLAY** = 'married'

			case marital_status_cd is NULL then **CODE** = 'UNK' and **DISPLAY** = 'unknown'

statecityzip_path, zip_cd	Patient.address	Patient	*select split_part(statecityzip_path, '\', 2) as STATE,
			split_part(statecityzip_path, '\', 3) as CITY, zip_cd from public.patient_dimension where patient_num = patient_num
			(Zip codes\Massachusetts\Cambridge\02140\)
			{
			"use" : "home" ,
			"type" : "both",
			"city" : **CITY**
			"state" : **STATE**
			"postalCode" : **zip_cd**
			"country" : "US"
			}