INFORMATICS PROFESSIONALS. LEADING THE WAY.

DataKnots: A Framework for Building Domain Specific Query Languages

S02: Systems Demonstrations - Bringing it all together: The best of data integration

Clark C. Evans

The Mechanical Rabbit Collaboration

Twitter: @clarkevans

#IS21

Disclosure

I disclose the following relevant relationship with commercial interests:

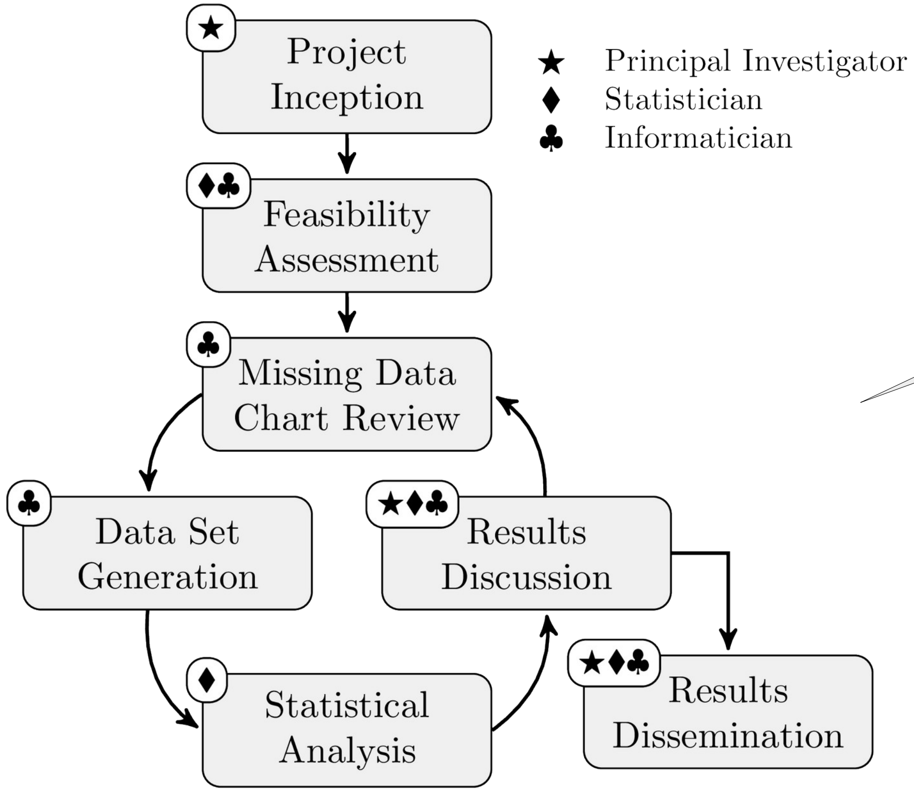
- previously, technical co-founder of Prometheus Research, an IQVIA business

Learning Objectives

After participating in this session the learner should be better able to:

- understand how Query Combinators provides formal semantics for query tools, encapsulating technical detail with a vocabulary relevant to the physician scientist;
- learn how our Julia implementation, Data Knots, can be used to generate domain specific query languages, usable at multiple levels of software construction; and
- consider database queries as a formal notation for research inquiries, enabling multidisciplinary team collaboration.

Research Collaboration Workflow



data-sets facilitate knowledge sharing

Diagram from “A centralized research data repository enhances retrospective outcomes research capacity: A case report”, William Hruby, et all. JAMIA 2013

Domain Specific Query Languages (DSQL)

```
# Clinical Quality Language implementation of CQL 124v7

library CervicalCancerScreening version '7.2.000'
using QDM version '5.3'
include MATGlobalCommonFunctions version '2.0.000' called Global
⋮
valueset "Pap Test": 'urn:oid:2.16.840.1.113883.3.464.1003.108.12.1017'
⋮
define "Pap Test with Results":
  ["Laboratory Test, Performed": "Pap Test"] PapTest
   where PapTest.result is not null
⋮
define "PapTest Within 5 Years":
  ("Pap Test with Results" PapTestOver30YearsOld
     with ["Patient Characteristic Birthdate"] BirthDate
       such that Global."CalendarAgeInYearsAt"(
                   BirthDate.birthDatetime,
                   start of PapTestOver30YearsOld.relevantPeriod)>= 30
         and PapTestOver30YearsOld.relevantPeriod 5 years
           or less before end of "Measurement Period"
        )
```

queries facilitate knowledge sharing

https://ecqi.healthit.gov/sites/default/files/ecqm/measures/CMS124v7.html

Query Anatomy : Data Model & Operations

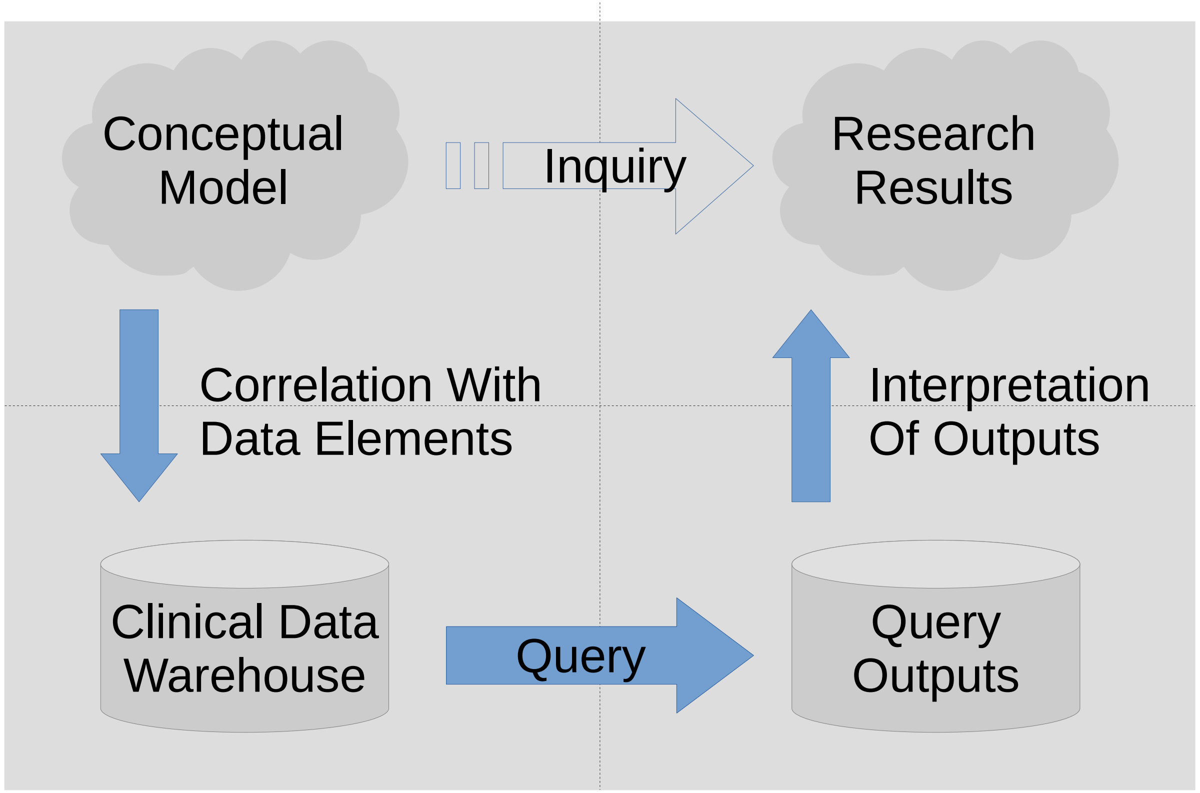
```
# Clinical Quality Language implementation of CQL 124v7

library CervicalCancerScreening version '7.2.000'
using QDM version '5.3'
include MATGlobalCommonFunctions version '2.0.000' called Global
⋮
valueset "Pap Test": 'urn:oid:2.16.840.1.113883.3.464.1003.108.12.1017'
⋮
define "Pap Test with Results":
  ["Laboratory Test, Performed": "Pap Test"] PapTest
   where PapTest.result is not null
⋮
define "PapTest Within 5 Years":
  ("Pap Test with Results" PapTestOver30YearsOld
     with ["Patient Characteristic Birthdate"] BirthDate
       such that Global."CalendarInYearsAt"(
                   BirthDate.birthDateTime,
                   start of PapTestOver30YearsOld.relevantPeriod)>= 30
         and PapTestOver30YearsOld.relevantPeriod 5 years
           or less before end of "Measurement Period"
        )
```

vocabulary & logic accessible to domain experts

```
Boilerplate
Variable Bindings
Data Model
Core Operations
Extensions
```

https://ecqi.healthit.gov/sites/default/files/ecqm/measures/CMS124v7.html

Conceptual Inquiry and Executable Query



For collaborations, queries should describe the inquiry

DataKnots.jl : A Framework for DSQLs

- *Tight Relationship Between Conceptual Inquiries & Executable Queries*
- Vocabulary reflecting relevant data sources and computations
- Query systems as ecosystem of interoperable modules
- Integration of multiple data sources within a single query
- Shared logic and syntax permit construction of great tooling
- Tailored can evolve, to support incremental discovery

https://github.com/MechanicalRabbit/DataKnots.jl/

DataKnots Prototype: Quality Measures

```
module CMS124
using DataKnots
using DataKnots4FHIR # for data model and extensions
⋮
@valueset PapTest = "2.16.840.1.113883.3.464.1003.108.12.1017"
⋮
@define PapTestWithResults =
          LaboratoryTestPerformed.
            filter(code.matches(PapTest) && exists(value))

⋮
@define PapTestWithin5Years =
          let birthDate => PatientCharacteristicBirthdate.birthDateTime,
              previous5years => MeasurePeriod.end.and_previous(5years),
              ageAtTest => years_between(relevantPeriod.start, birthDate)
            PapTestWithResults.
            filter(ageAtTest >= 30 && relevantPeriod.during(previous5years))
          end
⋮
end
```

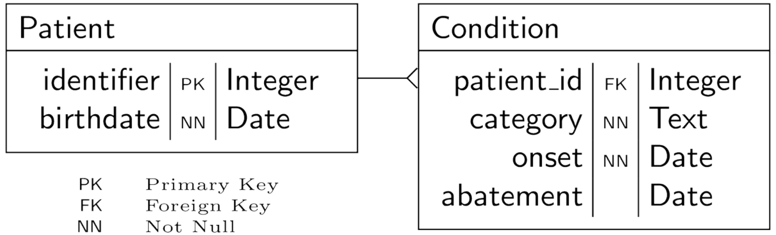
vocabulary & logic accessible to domain experts

```
Boilerplate
Variable Bindings
Data Model
Core Operations
Extensions
```

https://github.com/MechanicalRabbit/DataKnots4FHIR.jl/blob/master/doc/src/cms124v7.jl

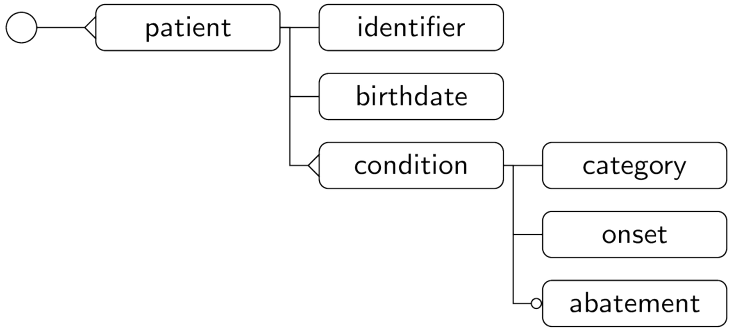
Query Primitives

Tabular Model

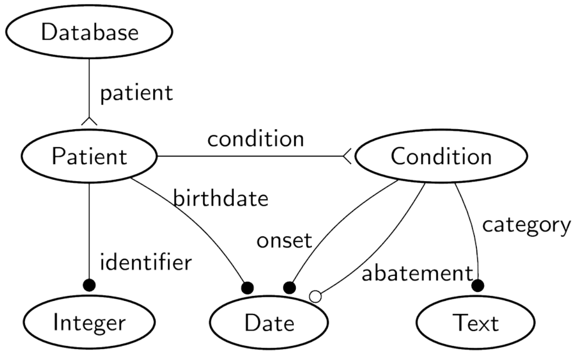


PK	Primary Key
FK	Foreign Key
NN	Not Null

Hierarchical Model



Functional Model



Primitive Queries

Primitive	Signature
patient	Database $\rightarrow$ Patient*
identifier	Patient $\rightarrow$ Integer
birthdate	Patient $\rightarrow$ Date
condition	Patient $\rightarrow$ Condition*
category	Condition $\rightarrow$ Text
onset	Condition $\rightarrow$ Date
abatement	Condition $\rightarrow$ Date$^?$

Query Combinators

Count Combinator

$$\begin{array}{rl} f & A \to B^* \\ \hline \mathsf{count}(f) & A \to \mathsf{Integer} \end{array}$$

count() applied to patient

$$\begin{array}{rl} \mathsf{patient} & \mathsf{Database} \to \mathsf{Patient}^* \\ \hline \mathsf{count}(\mathsf{patient}) & \mathsf{Database} \to \mathsf{Integer} \end{array}$$

Composition Combinator

$$\begin{array}{rl} f & A \to B^* \\ g & B \to C^* \\ \hline f.g & A \to C^* \end{array}$$

Composition of patient and condition

$$\begin{array}{rl} \mathsf{patient} & \mathsf{Database} \to \mathsf{Patient}^* \\ \mathsf{condition} & \mathsf{Patient} \to \mathsf{Condition}^* \\ \hline \mathsf{patient.condition} & \mathsf{Database} \to \mathsf{Condition}^* \end{array}$$

Encapsulate Technical Query Details

How many patients, 18 or older, have an active diagnosis of Essential Hypertension?

Adults With Hypertension

```
patient
filter    $($age $>=$ 18   &&
              has_active_diagnosis(
                        essential_hypertension))
count()
```

Domain Definitions

```
essential_hypertension $=$ "59621000"
age $=$ years(now() $-$ birthdate)
has_active_diagnosis($x$) $=$
        exists(condition.filter(
                  category $==$ $x$   &&
                  is_null(abatement)))
```

Query Fragment, Revisited

```
module CMS124
using DataKnots
using DataKnots4FHIR # for data model and extensions
⋮
@valueset PapTest = "2.16.840.1.113883.3.464.1003.108.12.1017"
⋮
@define PapTestWithResults =
          LaboratoryTestPerformed.
            filter(code.matches(PapTest) && exists(value))

⋮
@define PapTestWithin5Years =
          let birthDate => PatientCharacteristicBirthdate.birthDateTime,
              previous5years => MeasurePeriod.end.and_previous(5years),
              ageAtTest => years_between(relevantPeriod.start, birthDate)
            PapTestWithResults.
            filter(ageAtTest >= 30 && relevantPeriod.during(previous5years))
          end
⋮
end
```

Combinator	Definition
matches	Valueset matching on a domain.
and_previous	Extends a time interval to the left.
years_between	Compares years between dates
during	If an interval (or date) occurs within another

```
Boilerplate
Variable Bindings
Data Model
Core Operations
Extensions
```

https://github.com/MechanicalRabbit/DataKnots4FHIR.jl/blob/master/doc/src/cms124v7.jl

Julia Syntax (expanding @define macro)

```
# DataKnots -- Macro Syntax

@define PapTestWithin5Years =
          let birthDate => PatientCharacteristicBirthdate.birthDateTime,
              previous5years => MeasurePeriod.end.and_previous(5years),
              ageAtTest => years_between(relevantPeriod.start, birthDate)
            PapTestWithResults.
            filter(ageAtTest >= 30 && relevantPeriod.during(previous5years))
          end


# DataKnots -- Native Julia Syntax

PapTestWithin5Years =
Given(:birthDate => It.PatientCharacteristicBirthdate >> It.birthDateTime,
      :previous5years => and_previous.(It.MeasurePeriod >> It.end, years(5)),
      :ageAtTest => years_between.(It.relevantPeriod >> It.start, It.birthDate),
  It.PapTestWithResults >>
  Filter((It.ageAtTest .>= 30) .& during.(It.relevantPeriod, It.previous5years)) >>
  Label(:PapTestWithin5Years)
```

Julia native syntax is identical, but a bit more noisy

The Quality Data Model Layer

The **years_between** combinator is a Julia function lifted to queries:

```
years_between(lhs::Date, rhs::Date) =
    year(lhs) - year(rhs) -
     (month(lhs) > month(rhs) ? 0 :
     (month(lhs) < month(rhs) ? 1 :
     (day(lhs) >= day(rhs) ? 0 : 1)))
```

The **PatientCharacteristicBirthdate** is QDM element is also a query:

```
FHIR_Patient = FHIRProfile(:STU3, "Patient")

QDM_PatientCharacteristicBirthdate =
    It.entry >> It.resource >> FHIR_Patient >> Is1to1 >>
    Record(
     :code => Set{Coding}([Coding(:LOINC, Symbol("21112-8"))]),
     :birthDateTime => DateTime.(It.birthDate) >> Is1to1
    ) >> Label(:PatientCharacteristicBirthdate)
```

The QDM layer is lifted functions and simple queries

https://github.com/MechanicalRabbit/DataKnots4FHIR.jl/blob/master/src/quality.jl#L30

The FHIR Profile Layer

First, we unpack the profiles by reading FHIR Profile JSON…

```
    UnpackProfile =
      Record(
        It.id >> IsString,
        :kind => Symbol.(It.kind >> IsString),
        :elements =>
          It.snapshot >> IsDict >>
          It.element >> IsVector >> IsDict >>
          Filter((It.max >> IsString) .!= "0") >>
       Record(…) >> Drop(1))
```

Then, `FHIRProfile(:STU3, "Patient")` dynamically constructs a query.

```
    function FHIRProfile(standard::Symbol, profile)
        …
        return IsDict >> resourceFilter >>
            build_profile(…) >> Label(ident)
     end
```

The FHIR layer builds queries from profile definition

https://github.com/MechanicalRabbit/DataKnots4FHIR.jl/blob/master/src/profile.jl#L54

Comparison of DataKnots and CQL

```
# Clinical Quality Language implementation of CQL 124v7

define "PapTest Within 5 Years":
  ("Pap Test with Results" PapTestOver30YearsOld
     with ["Patient Characteristic Birthdate"] BirthDate
       such that Global."CalendarInYearsAt"(
                   BirthDate.birthDateTime,
                   start of PapTestOver30YearsOld.relevantPeriod)>= 30
         and PapTestOver30YearsOld.relevantPeriod 5 years
           or less before end of "Measurement Period"
        )


# DataKnots – CQL Inspired Domain Specific Query Language

@define PapTestWithin5Years =
          let birthDate => PatientCharacteristicBirthdate.birthDateTime,
              previous5years => MeasurePeriod.end.and_previous(5years),
              ageAtTest => years_between(relevantPeriod.start, birthDate)
            PapTestWithResults.
            filter(ageAtTest >= 30 && relevantPeriod.during(previous5years))
          end
```

DataKnots is comparable but also generalizable

- local variables
- regular syntax
- clean semantics
- domain specific operations

SQL Alternative for OHDSI Cohort Queries

Occurrences of an angioedmia diagnosis during an inpatient or emergency room visit

(from The Book of OHDSI, chapter 9: SQL and R)

```
person
keep(erip_visit => visit.filter(
    concept.iscoded("Visit",
                    "ERIP", "ER", "IP"))
condition
keep(index_date => start_date)
filter(concept.iscoded("SNOMED", 41291007))
filter(erip_visit.includes(index_date))
{ subject_id => person.person_id,
  cohort_start_date => start_date,
  cohort_end_date => end_date }
```

```
SELECT cohort_start_date, cohort_end_date, subject_id
FROM (
  SELECT DISTINCT person_id AS subject_id,
    condition_start_date AS cohort_start_date,
    condition_end_date AS cohort_end_date
  FROM @cdm_db_schema.condition_occurrence
  INNER JOIN concept_ancestor
    ON condition_concept_id = descendant_concept_id
  WHERE ancestor_concept_id = 432791
) distinct_occurrence
INNER JOIN schema.visit_occurrence
  ON subject_id = person_id
  AND visit_start_date <= cohort_start_date
  AND visit_end_date >= cohort_start_date
WHERE visit_concept_id IN (262, 9203, 9201);
```

https://github.com/MechanicalRabbit/ohdsi-synpuf-demo/blob/master/cohorts/1770674.md

Implementation Notes & Performance

- Given `JSON.jl` and DataKnots.jl, implementation took 27 days
- Maintainable encapsulation of logic at FHIR, QDM, and eCQM layers
- Computation of CMS124 over 1,000 patients averages 76ms per patient with a single core on a i7-4770 desktop computer
- Computation of same measure using Database Consulting Group, CQL Measure Processing Component (https://github.com/DBCG/cqf-ruler) on same computer averaged 607ms per patient
- Our bottleneck is memory usage, with a high-water mark of 11mb per patient. JSON parsing is expensive (17ms/patient)
- Next step is push-down of query logic into query specific JSON parser

https://github.com/MechanicalRabbit/DataKnots4FHIR.jl/blob/master/test/benchmark.jl

INFORMATICS PROFESSIONALS. LEADING THE WAY.

Thank you!

Email me at:
info@clarkevans.com