Query Combinators for Medical Research and Decision Support

an algebraic theory of database queries applied to medicine

Clark C. Evans ⟨cce@clarkevans.com⟩,
Kyrylo Simonov ⟨xi@resolvent.net⟩

Tuesday, May 28, 2019
OHSDI Community Meeting

Prometheus Research, LLC

Outline of Talk

1. Introduction
2. Example: Complex Query
3. Thinking in Query Combinators
4. Example: Feasibility Assessment
5. Conclusion

Introduction

Clinical Research Workflow

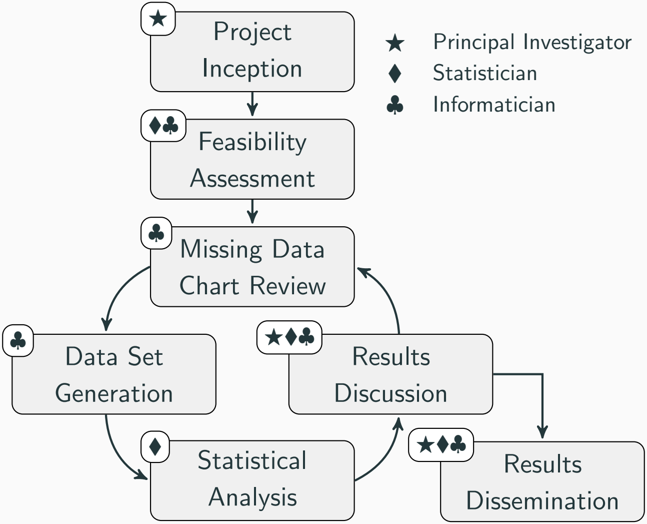
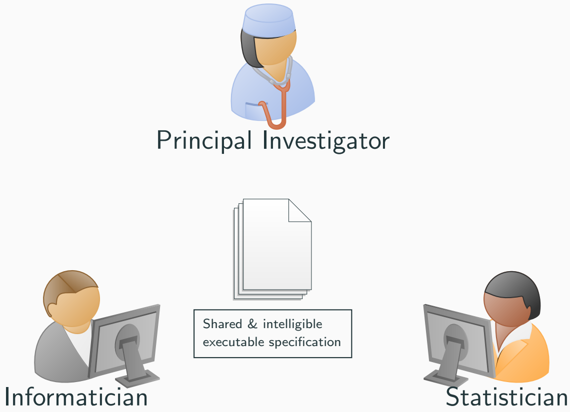


Figure 1: Clinical Research Workflow as inspired from Hruby's observations at Columbia University [2]

Current Practice: Multiple Query Languages

Is Shared Query Infrastructure Is Possible?



Example: Complex Query

Example: Complex Query

Consider the inquiry, "Which anti-hypertensive medications are effective in improving blood pressure?". This inquiry could be operationalized as:

Within 6 months of a hypertension diagnosis, when an anti-hypertensive medication was added or intensified, was there a blood pressure decrease of 5 mmHg or more within 5 days after the medication adjustment?

Example: Complex Query

Consider the inquiry, "Which anti-hypertensive medications are effective in improving blood pressure?". This inquiry could be operationalized as:

> *Within 6 months of a* **hypertension diagnosis**, *when an* **anti-hypertensive medication** *was* **added or intensified**, *was there a* **blood pressure decrease** *of 5 mmHg or more within 5 days after the* **medication adjustment***?*

What are the query components?

The first thing to do is convert specialized vocabulary in this inquiry into query component definitions in a *query mediation* session.

Component	Mediation Notes
hypertension_diagnosis	exclude pregnancy & kidney failure
antihypertensive_medication	a product list is provided
added_or_intensified	new therapy or larger dose
blood_pressure_decrease	of both systolic & diastolic
medication_adjustment	change of daily medication
active_ingredient	normalize dosage records across compound products

Table 1: Anti-hypertensive Query Components

Anti-Hypertensive Query

```
patient.keep(it)
antihypertensive_medication
active_ingredient
medication_adjustment
filter(added_or_intensified &
      during(previous(6months), patient.hypertension_diagnosis)
collect(is_effective =>
      during(subsequent(5days),
         patient.blood_pressure_decrease(5mmHg)))
group(active_ingredient)
{ active_ingredient,
   count(medication_adjustment.filter(is_effective)),
   count(medication_adjustment.filter(not(is_effective))) }
```

Query Elements and Operations

This query brings together many things, including:

- query composition algebra;
- built-in combinators, such as filter, collect, group, keep, count, etc.;
- data source queries, including patient and medication;
- domain specific queries, such as medication_adjustment, active_ingredient, and blood_pressure_decrease; and
- domain specific combinators, such as during and subsequent;

The domain specific queries and combinators are then independently defined, constructed, documented, and tested. They can be reused across questions and reflect the shared vocabulary for the research team.

Thinking in Query Combinators

Tabular Model of Clinical Research Data Repository

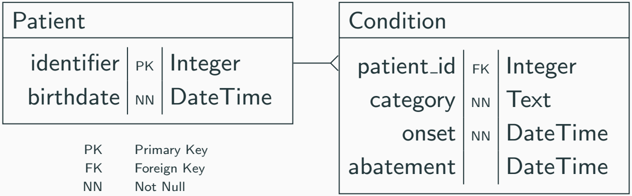


Figure 2: Tabular Model for CRDR

Hierarchical Model of Clinical Research Data Repository

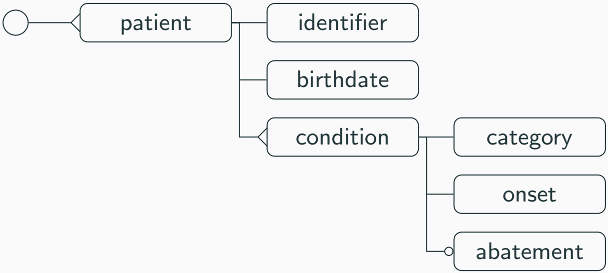
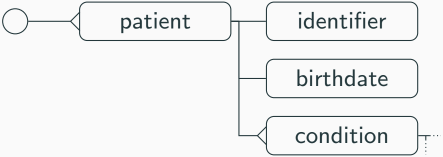


Figure 3: Hierarchical Model for CRDR

Example Queries



- patient
- count(patient)
- patient.condition
- patient.count(condition)
- mean(patient.count(condition))

Query Combinator Algebra

Query Combinators are an algebra of query functions.

- This algebra's elements, or *queries*, represent relationships among class entities and datatypes.
- This algebra's operations, or *combinators*, are applied to construct query expressions.

Query expressions, such as count(condition) are constructed by applying combinators, such as count to queries, such as condition.

Functional Model

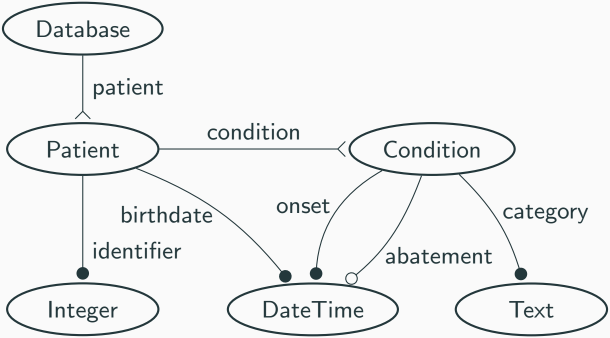


Figure 4: Functional Model for CRDR

Query Primitives

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

Table 2: Query Primitives for CRDR

The Count Combinator

$$\frac{f \qquad A \to B^*}{\text{count}(f) \qquad A \to \text{Integer}}$$

$$\frac{\text{patient} \qquad \text{Database} \to \text{Patient}^*}{\text{count(patient)} \qquad \text{Database} \to \text{Integer}}$$

$$\frac{\text{condition} \qquad \text{Patient} \to \text{Condition}^*}{\text{count(condition)} \qquad \text{Patient} \to \text{Integer}}$$

The Composition Combinator

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

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

$$\frac{\begin{array}{rl} \text{condition} & \text{Patient} \to \text{Condition}^* \\ \text{category} & \text{Condition} \to \text{Text}^* \end{array}}{\begin{array}{rl} \text{condition.category} & \text{Patient} \to \text{Text}^* \end{array}}$$

Example: Feasibility Assessment

Example: Feasibility Assessment

Suppose that an informatician would like to conduct a feasibility assessment to see if the CRDR database has at least some candidate patients relevant to this hypertension effectiveness inquiry.

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

Components of Feasibility Assessment

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

Component	Definition
essential_hypertension	`'59621000'`
age	years(now() − birthdate)
has_active_diagnosis(x)	exists(condition.filter(category $= x$ & is_null(abatement)))

Table 3: Component Definitions for Feasibility Assessment

Adults /w Hypertension

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

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

Conclusion

There is an Implementation: DataKnots.jl

There is an implementation of Query Combinators for the Julia Language, called `DataKnots.jl`.

- this implementation is MIT/Apache licensed
- it includes an in-memory, column-oriented database
- it has adapters to CSV (and soon XML, JSON)
- essential query operators are implemented
- Julia statistics can be *lifted* to a combinator
- an adapter to SQL datasources is in progress!

https://github.com/rbt-lang/DataKnots.jl

Credits

- Thanks to James Shalaby, Pharm.D. for his hypertension research question and for query mediation discussion.
- Thanks to Simons Foundation for years of funding for earlier variants of this initative, called HTSQL.

References i

C. C. Evans and K. Simonov.
Query combinators.
2017.

G. Hruby, J. McKiernan, S. Bakken, and C. Weng.
A centralized research data repository enhances retrospective outcomes research capacity: A case report.
20, 01 2013.