The Constellation Query Language

A semantic modeling and query language

Clifford Heath
Who am I

• Software product designer (HP then start-ups)
Who am I

• Software product designer (HP then start-ups)
• 30 years in the software industry
Who am I

• Software product designer (HP then start-ups)
• 30 years in the software industry
• S/W Development tools and languages
Who am I

• Software product designer (HP then start-ups)
• 30 years in the software industry
• S/W Development tools and languages
• Software deployment products (ManageSoft)
Who am I

- Software product designer (HP then start-ups)
- 30 years in the software industry
- S/W Development tools and languages
- Software deployment products (ManageSoft)
- Each product larger than 1 million LOC
Who am I

• Software product designer (HP then start-ups)
• 30 years in the software industry
• S/W Development tools and languages
• Software deployment products (ManageSoft)
• Each product larger than 1 million LOC
• Large enterprises, products and deployments
Who am I

• Software product designer (HP then start-ups)
• 30 years in the software industry
• S/W Development tools and languages
• Software deployment products (ManageSoft)
• Each product larger than 1 million LOC
• Large enterprises, products and deployments
• Manufacturers, banks, telcos, governments
Who am I

- Software product designer (HP then start-ups)
- 30 years in the software industry
- S/W Development tools and languages
- Software deployment products (ManageSoft)
- Each product larger than 1 million LOC
- Large enterprises, products and deployments
- Manufacturers, banks, telcos, governments
- Millions of corporate desktops and servers
State of the RDBMS art
State of the RDBMS art
State of the RDBMS art
ActiveFacts project

A project of Data Constellation

- Constellation Query Language (CQL)
ActiveFacts project

A project of Data Constellation

- Constellation Query Language (CQL)
- Generates to SQL and O-O class libraries
ActiveFacts project

A project of Data Constellation

-Constellation Query Language (CQL)
  -Generates to SQL and O-O class libraries
-Constellation API
ActiveFacts project

A project of Data Constellation

- Constellation Query Language (CQL)
  - Generates to SQL and O-O class libraries
- Constellation API
  - A new programming approach for data
ActiveFacts project

A project of Data Constellation

• Constellation Query Language (CQL)
  • Generates to SQL and O-O class libraries
• Constellation API
  • A new programming approach for data
• Graphical modeling tools (APRIMO)
ActiveFacts project

A project of Data Constellation

- Constellation Query Language (CQL)
  - Generates to SQL and O-O class libraries
- Constellation API
  - A new programming approach for data
- Graphical modeling tools (APRIMO)
  - Online collaborative modeling
ActiveFacts project

A project of Data Constellation

• Constellation Query Language (CQL)
  • Generates to SQL and O-O class libraries
• Constellation API
  • A new programming approach for data
• Graphical modeling tools (APRIMO)
  • Online collaborative modeling
Goals
of the Constellation Query Language

• Include all capabilities of ORM2
Goals

of the Constellation Query Language

• Include all capabilities of ORM2
• Use plain text with minimal markup & math
Goals

of the Constellation Query Language

• Include all capabilities of ORM2
• Use plain text with minimal markup & math
• As close to natural language as possible
Goals
of the Constellation Query Language

• Include all capabilities of ORM2
• Use plain text with minimal markup & math
• As close to natural language as possible
• Strictly limit the use of reserved phrases
Goals

of the Constellation Query Language

• Include all capabilities of ORM2
• Use plain text with minimal markup & math
• As close to natural language as possible
• Strictly limit the use of reserved phrases
• Express queries able to surpass SQL
Goals

of the Constellation Query Language

• Include all capabilities of ORM2
• Use plain text with minimal markup & math
• As close to natural language as possible
• Strictly limit the use of reserved phrases
• Express queries able to surpass SQL
• Support non-English derivatives
Goals
of the Constellation Query Language

• Include all capabilities of ORM²
• Use plain text with minimal markup & math
• As close to natural language as possible
• Strictly limit the use of reserved phrases
• Express queries able to surpass SQL
• Support non-English derivatives
• Generate both object and relational code
Simple ORM2 Example

This and all other ORM2 examples were created using NORMA, the Natural ORM2 Architect
Relational model

**Person**
- BirthDate
- EmployeeCompany_Name
- EmployeeManagerNr
- EmployeeNr
- FamilyName
- GivenName
- ManagerIsCeo

**Directorship**
- AppointmentDate
- Company_Name
- DirectorFamilyName
- DirectorGivenName

**Attendance**
- AttendeeFamilyName
- AttendeeGivenName
- MeetingCompany_Name
- MeetingDate
- MeetingIsBoardMeeting

**Company**
- Company_Name
- IsListed

Relationships:
- Person → Works at → Company
- Person → Employs → Attendance
- Directorship → DirectorFamilyName
- EmployeeManagerNr
- EmployeeNr
- Attendance → MeetingIsBoardMeeting
- Company → IsListed
Vocabulary

- Every CQL declaration is in a Vocabulary
- Not “Schema” or “Model”, though it is both
- White space is not significant
- Comments conform to C++ style
Value Types

ORM2:

Company Name is written as String(48);
Employee Nr is written as Integer;
Course is written as String(2) restricted to {'A'..'E', 'PW'};

Each definition creates or implies two ValueTypes (no Data Types)
ValueTypes parameters (Length and Scale) are not yet user-extendable
Terms

- Case sensitive
- Single word (soon to be multiple)
- Normally in TitleCase (that’s not mandatory)
- A term may have synonyms in this or other vocabularies
- All colours used for illustration only:
  - Terms are mauve
  - Blue for reserved words
  - Orange for values and units
  - Green for other words
Unit conversions

9.80665 m sec^-2 converts to gravity;
5/9 degC + 32 converts to degF;

A library of more than 500 conversions and constants is available.

1 newton m converts to joule / joules;
1055.06 joule converts to britishthermalunit approximately;
0.8945 usdollar converts to australiadollar ephemeral;
Unit conversions

9.80665 m sec^-2 converts to gravity;
5/9 degC + 32 converts to degF;

A library of more than 500 conversions and constants is available.

1 newton m converts to joule / joules;
1055.06 joule converts to britishthermalunit approximately;
0.8945 usdollar converts to australiadollar ephemeral;

Conversion between compatible units is automatic. Incompatible conversions are rejected. Any unit having no conversion is assumed fundamental.
Fact Types

ORM2:

CQL:

```cql
Person was born on birth-Date;
```
Fact Types

ORM2:

CQL:

Person was born on at most one birth-Date;
Fact Types

ORM2:

CQL:

Person was born on birth-Date restricted to \{'1900/01/01' .. \};
Fact Types

ORM2:

CQL:

Person was born on birth-Date restricted to { '1900/01/01' .. };

Person was born at at most one birth-Place;
Fact Types

ORM2:

CQL:

Person was born on birth-Date restricted to { '1900/01/01' .. };

Person was born at at most one birth-Place;

Meeting is board meeting;
Entity Types

ORM2:

Company (.Name)

CQL:

Company is identified by its Name;

Means the same as ...
Entity Types

ORM2:

Company

is called / is of

CompanyName

CQL:

Company is identified by its Name;

Means the same as ...

CompanyName is written as Name;
Company is identified by CompanyName where Company has one CompanyName, CompanyName is of at most one Company;
Entity Types

ORM2:

```
  Company
  \  /   \   \    \\
   v     v     v  \\
 CompanyName
```
is called / is of

CQL:

```
Company is identified by its Name;
```

Means the same as ...

```
CompanyName is written as Name;
Company is identified by CompanyName where
Company has one CompanyName,
CompanyName is of at most one Company;
```

Custom reading:

```
Company is identified by its Name where
Company is called CompanyName;
```
Claim concerns at most one Incident, Incident is of one Claim;
One-to-one fact types

Claim concerns at most one Incident, Incident is of one Claim;

Two uniqueness constraints require two readings
One-to-one fact types

Incident is identified by Claim where Claim concerns at most one Incident, Incident is of one Claim;
Composite Identification

ORM2:

CQL:

Person is identified by given Name and family Name where
Person has one given-Name,
given Name is of Person,
family-Name is of Person,
Person is called at most one family Name;

...Two Fact Types, each with 2 readings!
Hyphens - new

- An un-spaced hyphen between two non-terms is ignored, e.g. semi-trailer
- A hyphen between a term and a non-term (adjective) introduces a new local term
  - e.g. given- Name, Nom-donné
- A hyphen with one adjacent space introduces a local term (with perhaps > 2 words)
  - e.g. original- tax Amount
- Local terms need only one introduction
- In other cases, a hyphen means arithmetic minus
  - e.g. between two terms, or before digits
- Local terms are used in binding fact types
Subtypes

ORM2:

CQL:

Manager is a kind of Employee;
Subtypes

ORM2:

CQL:

Employee is a kind of Person identified by its Nr;
Manager is a kind of Employee;

Multiple supertypes are allowed
Subtypes

ORM2:

CQL:

Employee is a kind of Person identified by its Nr;
Manager is a kind of Employee;

Employee is supervised by at most one Manager [acyclic];
Manager is ceo;
Objectified Fact Types

ORM2:

CQL:

Directorship is where
Person directs Company,
Company is directed by at least one Person;
Objectified Fact Types

"Directorship"

directs / is directed by

Directorship is where
   Person (as Director) directs Company,
   Company is directed by at least one Director;
Objectified Fact Types

"Directorship"

ORM2:

directs / is directed by

CQL:

Directorship is where
  Person (as Director) directs Company,
  Company is directed by at least one Director;

and then we can:

Directorship began on one appointment-Date;
Objectified Fact Types

"Directorship (.Id)"

CQL:

Directorship is identified by its Id where
Person (as Director) directs Company,
Company is directed by at least one Director;

External identification
Ternary and higher

StudentParticipation is where
Student represent School in Activity,
Student participates in Activity sanctioned by at most one School;
Ternary and higher

StudentParticipation is where

Student represent School in Activity,
Student participates in Activity sanctioned by at most one School;

subset constraint:

Student represents School in Activity
only if School sanctions Activity;
Mandatory / Exclusive

either Employee is ceo
or Employee is supervised by Manager but not both;

Mandatory and exclusive
for each Employee exactly one of these holds:
that Employee is CEO,
that Employee is supervised by some Manager;

Mandatory and exclusive
Mandatory / Exclusive

for each Employee exactly one of these holds:
that Employee is CEO,
that Employee is supervised by some Manager;

Mandatory and exclusive

Also mandatory non-exclusive
Exclusive non-mandatory

(at least one)
(at most one)
Equivalence

Event is in Series if and only if Event has Number;

Events which are in a series must all have a number and vice versa.
Join constraints

PurchaseOrderItem matches SalesOrderItem only if PurchaseOrderItem is for Product and SalesOrderItem is for Product;
Join constraints

**PurchaseOrderItem** matches **SalesOrderItem** only if **PurchaseOrderItem** is for **Product** and **SalesOrderItem** is for **Product**;

**Product** is the same instance in both clauses
Join constraints

Product is the same instance in both clauses

In theory, any number of clauses can be joined (the join path may be of any length)

PurchaseOrderItem matches SalesOrderItem only if PurchaseOrderItem is for Product and SalesOrderItem is for Product;
Fact Instances

Value Type: Name 'Microsoft';
Entity Type: Company 'Google';
Composite: given Name 'Fred' is of Person, Person was born at birth Place 'Ballarat';

... note the join over Person

Join Contraction:
given Name 'Fred' is of Person who was born at birth Place 'Ballarat';

... the contracted form isn’t implemented yet!
Joins

- Joins occur when 2+ role references are bound
  - Multiple bindings are possible using adjectives or role names
  - Subscripts are a fall-back

- Subtyping joins may be implicit (if unambiguous):
  Employee is ceo, not only Manager is ceo

- Objectification joins (new syntax, not implemented):
  Directorship (where Person directs Company) began on appointment-Date, Company is listed
Insurance Fact Types

Driver is a kind of Person;

VehicleIncident is a kind of Incident;

Driving is where Driver drove vehicle in VehicleIncident;

VehicleIncident followed Intoxication;

DrivingCharge is where Driving resulted in Charge;

DrivingCharge is a warning;

These definitions support the join example that follows.
Join Types

Population name

dexample:
given Name 'Barry' is of Person who has family Name 'Smith' and Driving (where Person drove vehicle in Incident which is of Claim 4738) followed Intoxication 'marijuana' and DrivingCharge (where Driving resulted in Charge 'driving under the influence') is a warning;

Fact type readings are taken from the Insurance example at http://dataconstellation.com
Join Types

Ordinary joins

element: given Name 'Barry' is of Person who has family Name 'Smith' and Driving (where Person drove vehicle in Incident which is of Claim 4738) followed Intoxication 'marijuana' and DrivingCharge (where Driving resulted in Charge 'driving under the influence') is a warning;

Ordinary joins can also use comma, except in some constraints.

Fact type readings are taken from the Insurance example at http://dataconstellation.com
Join Types

Ordinary joins

eexample:
given Name 'Barry' is of Person who has family Name 'Smith' and Driving (where Person drove vehicle in Incident which is of Claim 4738) followed Intoxication 'marijuana' and DrivingCharge (where Driving resulted in Charge 'driving under the influence') is a warning;

Ordinary joins can also use comma, except in some constraints.

Fact type readings are taken from the Insurance example at http://dataconstellation.com
Join Types

Ordinary joins

example:
given Name 'Barry' is of Person who has family Name 'Smith' and Driving (where Person drove vehicle in Incident which is of Claim 4738) followed Intoxication 'marijuana' and DrivingCharge (where Driving resulted in Charge 'driving under the influence') is a warning;

Ordinary joins can also use comma, except in some constraints.

Fact type readings are taken from the Insurance example at http://dataconstellation.com
Join Types

example:
given Name 'Barry' is of Person who has family Name 'Smith' and Driving (where Person drove vehicle in Incident which is of Claim 4738) followed Intoxication 'marijuana' and DrivingCharge (where Driving resulted in Charge ‘driving under the influence’) is a warning;

Driving is where Driver drove vehicle in VehicleIncident;

Fact type readings are taken from the Insurance example at http://dataconstellation.com
Join Types

example:
given Name 'Barry' is of Person who has family Name 'Smith' and Driving (where Person drove vehicle in Incident which is of Claim 4738) followed Intoxication 'marijuana' and DrivingCharge (where Driving resulted in Charge 'driving under the influence') is a warning;

Value joins

Fact type readings are taken from the Insurance example at http://dataconstellation.com
Join Types

**Example:**
given Name 'Barry' is of Person who has family Name 'Smith' and Driving (where Person drove vehicle in Incident which is of Claim 4738) followed Intoxication 'marijuana' and DrivingCharge (where Driving resulted in Charge 'driving under the influence') is a warning;

Objectification joins

Fact type readings are taken from the Insurance example at http://dataconstellation.com
Join Types

Example:
given Name 'Barry' is of Person who has family Name 'Smith' and Driving (where Person drove vehicle in Incident which is of Claim 4738) followed Intoxication 'marijuana' and DrivingCharge (where Driving resulted in Charge 'driving under the influence') is a warning;

Contractions

Fact type readings are taken from the Insurance example at http://dataconstellation.com
Business Context Note

Person has one (as opposed to more than one, because we’ll join them into a single string, as agreed on 20 Aug 2009 by Bill, Jim) given-Name;

- Applies to constraint, fact type, object type
- A note with arbitrary text (matching parens)
- To record rationale, goal, or agreement
  - “so that”
  - “in order to”
  - “as opposed to”
  - “to avoid”
- “as agreed by”, “as agreed on <DATE> by”
Deontic Constraints

Event is in Series if and only if Event has Number
(otherwise email Organisers);
Man is married to at most one
(otherwise inform Hefner) Woman;

- Supported by all constraint types
- Identified by an Enforcement action
  - e.g. “alert”, “SMS”, “email”, “log”
- Action is applied in respect of an Agent
  - e.g. “security”, “auditors”, “Joe Smith”
Company is identified by its Name where
    Company is called CompanyName;
Company is listed;

Meeting is identified by Date and Meeting is board meeting and Company where
    Meeting held on one Date,
    Meeting is board meeting,
    Company held Meeting,
    Meeting is held by one Company;

Person is identified by given-Name and family-Name where
    Person has one given-Name,
    given-Name is of Person,
    family-Name is of Person,
    Person is called at most one family-Name;
Person was born on at most one birth-Date;
Attendance is where
    Person (as Attendee) attended Meeting,
    Meeting was attended by Attendee;
Directorship is where
    Person (as Director) directs Company,
    Company is directed by at least one Director;
Directorship began on one appointment-Date;

Employee is a kind of Person identified by its Nr;
Employee works at one Company,
    Company employs Employee;
Manager is a kind of Employee;
Employee is supervised by at most one Manager [acyclic],
    Manager supervises Employee;
Manager is ceo;
Queries

(work in progress)
Simple Query

Person has given Name ‘Daniel’?

... a contraction of the value join:

Person has given Name, given Name = ‘Daniel’?
Deriving Fact Types

Person directs Company ‘Acme, Inc’, Person has family Name?
Deriving Fact Types

family Name controls Company:
Company is directed by Person
who has family Name;
Deriving Fact Types

family Name controls Company: Company is directed by Person who has family Name;

family Name controls Company? family Name controls Company ‘Acme, Inc’? family Name ‘Heath’ controls Company?
Deriving Fact Types

family Name controls Company:
Company is directed by Person who has family Name;

family Name controls Company?
family Name controls Company ‘Acme, Inc’?
family Name ‘Heath’ controls Company?

Note that if a company has two directors from the same family, only one fact is derived.
Units conversion

Area is written as Real in mm\(^2\);

Pane has Area:
Pane of glass has Width,
Pane of glass has Height,
Width * Height = Area;

large Pane:
Pane has Area, Area >= 5 foot\(^2\);

large Pane?
“returning”

family Name controls Company:
Person directs Company,
Person has family Name,
returning Person,
by Company;

- “returning” makes Person.given_name available
- Causes result set to be non-first-normal-form (N1NF)
- “returning by” applies sorting
- Inner join semantics apply if family Name is unknown
“returning” is transitive

normal stuff for Person:
maybe Person was born on birth Date,
maybe Person is an Employee,
Employee has EmployeeNr,
returning birth Date, EmployeeNr;

normal stuff for Person
who is called family-Name ‘Smith’?

Doesn’t just return Person and family Name,
but also birth Date and EmployeeNr if known
Transitive queries

Employee works under Manager:
Employee is supervised by Manager [transitive];

Employee works under Manager,
Manager has given-Name ‘Joe’?
Transitive queries

Employee works under Manager:
Employee is supervised by Manager [transitive];

Employee works under Manager who has given-Name ‘Joe’?
either/or

family Name is associated with Company:
Person directs Company or
Person works at Company,
Person has family Name;
Date and Time

Person is adult:
Person was born on birth Date, birth Date \(<\) Now - 18 years;

or

Person is adult:
Person was born on birth Date which is before 18 years ago;
Aggregation functions

Company has SalaryBill:
SalaryBill (is sum of Salary in Employee receives Salary) and Employee works for Company;
Metamodel
Object Types

ValueType

is supertype of / is subtype of super-

nests / is nested as

TYPE INHERITANCE

is subtype of super- / is supertype of sub-

provides identification

EntityType

plays / is played by

Role

... has ... role

FactType (.Id)

Assimilation

{ 'separate', 'partitioned' }

Ordinal

Concept

is independent
Terms and Readings

Term synonyms are not shown.
Joins

{ 'neuter', 'personal', 'masculine', 'feminine' }

A join that traverses to or from an objectified fact type may lack one (phantom) role
Constraints

A deontic constraint is identified by its enforcement action.
Populations

- **Literal**: Lexical representation unless a string
- **Value**: represents/is represented by
- **Unit (.Id)**: is a string
- **Vocabulary (Name)**: includes/belongs to
- **Population**: includes/belongs to
- **Name**: is of/has
- **Example and Reference populations are standard**
- **Concept**: plays/is played by
- **Instance (.Id)**: includes/belongs to
- **RoleValue**: includes/fulfils
- **Fact (.Id)**: includes/belongs to
- **FactType (.Id)**: is of
- **ValueType**: is of
- **Fact**
Context

{ 'because', 'as_opposed_to', 'so_that', 'to_avoid' }

Concept

FactType (.Id)

Constraint (.Id)

ContextNote (.Id)

has / is of

Discussion

has / is for

Agent (.Name)

is according to / claims

Agreement

was added by / covers

was reached by / agreed on

Date

was on
CQL Generation from ORM₂
This and all other ORM2 examples were created using NORMA, the Natural ORM2 Architect
Generated CQL

Company is identified by its Name where
   Company is called CompanyName;
Company is listed;

Meeting is identified by Date and Meeting is board meeting and Company where
   Meeting held on one Date,
   Meeting is board meeting,
   Company held Meeting,
   Meeting is held by one Company;

Person is identified by given-Name and family-Name where
   Person has one given-Name,
   given-Name is of Person,
   family-Name is of Person,
   Person is called at most one family-Name;
Person was born on at most one birth-Date;
Attendance is where
   Person (as Attendee) attended Meeting,
   Meeting was attended by Attendee;
Directorship is where
   Person (as Director) directs Company,
   Company is directed by at least one Director;
Directorship began on one appointment-Date;

Employee is a kind of Person identified by its Nr;
Employee works at one Company,
   Company employs Employee;

Manager is a kind of Employee;
Employee is supervised by at most one Manager [acyclic],
   Manager supervises Employee;
Manager is ceo;
Relational Mapping
Mapping procedure

- Create a Reference (arc) for each many:1 and 1:1
  - This includes subtypes, which are 1:1
  - Binarize objectified fact types

- Mark tentative status for all object types
  - Value Types absorbed unless independent, Entity Types not.

- Iterate until no new decisions made:
  - Disregard existing final decisions, then
  - Objectified unaries are absorbed
  - An ET is always absorbed into an ET that identifies it
  - If ET has more than one reference to it and has functional dependencies that can’t absorb it, it’s a table (3NF)
  - If this object has one or more mandatory non-identifying roles that can absorb it, do that (this might flip some 1:1 references)
  - If this object has no functional dependencies apart from its identifying roles, it can be absorbed

- Finally a non-absorbed VT with FDs is a table
SQL Generation

Simplified: Foreign keys to nullable columns require an indexed VIEW
SQL Generation

Schema has been modified to suit a Rails application
CREATE TABLE Attendance (  
    AttendeeFamilyName  varchar(48) NULL,  
    AttendeeGivenName   varchar(48) NOT NULL,  
    MeetingCompanyName  varchar(48) NOT NULL,  
    MeetingDate         datetime NOT NULL,  
    MeetingIsBoardMeeting bit NOT NULL,  
    UNIQUE(AttendeeGivenName, AttendeeFamilyName, MeetingDate, MeetingIsBoardMeeting, MeetingCompanyName)  
)  
GO

CREATE TABLE Company (  
    CompanyName       varchar(48) NOT NULL,  
    IsListed          bit NOT NULL,  
    UNIQUE(CompanyName)  
)  
GO

CREATE TABLE Directorship (  
    CompanyName       varchar(48) NOT NULL,  
    DirectorFamilyName varchar(48) NULL,  
    DirectorGivenName  varchar(48) NOT NULL,  
    AppointmentDate    datetime NOT NULL,  
    UNIQUE(DirectorGivenName, DirectorFamilyName, CompanyName),  
    FOREIGN KEY(CompanyName) REFERENCES Company(CompanyName)  
)  
GO

CREATE TABLE Person (  
    FamilyName        varchar(48) NULL,  
    GivenName         varchar(48) NOT NULL,  
    BirthDate         datetime NULL,  
    EmployeeCompanyName varchar(48) NULL,  
    EmployeeManagerNr  int NULL,  
    EmployeeNr         int NULL,  
    ManagerIsCeo       bit NULL,  
    UNIQUE(GivenName, FamilyName)  
)  
GO
CREATE TABLE Directorship (  
  -- Directorship began on appointment-Date,  
  AppointmentDate       datetime NOT NULL,  
  -- Directorship is where Director directs Company and Company is called  
  CompanyName           varchar(48) NOT NULL,  
  -- Directorship is where Director directs Company and maybe family-Name  
  DirectorFamilyName    varchar(48) NULL,  
  -- Directorship is where Director directs Company and Person has given-  
  DirectorGivenName     varchar(48) NOT NULL,  
  UNIQUE(DirectorGivenName, DirectorFamilyName, CompanyName),  
  FOREIGN KEY (CompanyName) REFERENCES Company (CompanyName)  
)  
GO
Generated SQL - today

- Tables
- Columns
- Primary keys / Unique constraints
- Foreign Key constraints
- Indexes over views where needed
- CHECK constraints
Object-Oriented Mapping
Object-Oriented Mapping

- A Vocabulary becomes a module/package
- Objectified fact types are binarized
- Every object type becomes a class
- Every role has setter/getter (unary or array)
Constellation API

- **Constellation** is a population of fact instances over a **vocabulary**
- **No new/delete, just assert/deny**
- **Every instance** is of an object type
  - No raw values
- **No duplicate instances** in a constellation
- **Instances** are fully cross-referenced
  - Every role’s counterpart
Ruby API

- Uses meta programming
- Extends Ruby’s class/module metamodel
- Relationship methods create role pairs
- Implements multiple inheritance
- Supports verbalisation (though not yet using the readings)
module CompanyDirectorEmployee

  class CompanyName < String
    value_type :length => 48
  end

  class Company
    identified_by :company_name
    one_to_one :company_name, :mandatory => true # See CompanyName.company
    maybe :is_listed
  end

  class Person
    identified_by :given_name, :family_name
    has_one :birth_date, :class => Date # See Date.all_person_as_birth_date
    has_one :family_name, :class => Name # See Name.all_person_as_family_name
    has_one :given_name, :class => Name, :mandatory => true # See Name.all_person_as_given_name
  end

  class Directorship
    identified_by :director, :company
    has_one :company, :mandatory => true # See Company.all_directorship
    has_one :director, :class => Person, :mandatory => true # See Person.all_directorship
    has_one :appointment_date, :class => Date, :mandatory => true # See Date.all_directorship
  end

  class Employee < Person
    identified_by :employee_nr

  end

end
No need to generate the code

• Simply ‘require’ the CQL
• ActiveFacts generates and eval’s Ruby
• Open classes allow you to extend them
• No need to edit generated code
Persistence (not fully implemented yet)

• Ruby classes are correlated with tables through reflection of table names (only!)
• Column names are matched with roles
Persistence Model

- **One** query for each user action
- Each query yields a constellation
- Modify the constellation, then
- **One** save()
- Constraints enforced on save
- Changes are saved transactionally
Future work

- API Persistence and DBMS adapters
- Queries, including drag-drop GUI
- Other languages (natural and computer)
- APRIMO, a web-hosted semantic designer
- Reverse engineering
- Automatic migration
Demonstrations and Questions
Clifford Heath
Available for consulting and training

http://dataconstellation.com/