IDEAS 2021

Montreal, Canada

Categorical Management of Multi-Model Data

Martin Svoboda, Pavel Čontoš, Irena Holubová

svoboda@ksi.mff.cuni.cz

July 15, 2021

Charles University, Prague, Czech Republic

Data Variety

Structure of data

- **Logical models**
  - Relational, key/value, wide column, document, graph, …
- **Data formats**
  - XML or JSON for the document model, …
- **Schemas**
  - DTD or XML Schema schema languages, …
- **Vocabularies**
  - Names of XML elements or attributes, …

Other aspects

- **Technologies**: implementations, interfaces, protocols, …
- **Query languages**: syntax, constructs, expressive power

Database Systems

Traditional approach

- **Relational databases**
  - Primary option for decades
- Alternatives
  - Native XML databases, RDF stores, …

NoSQL databases

- Core models
  - Key/value, wide column, document, graph
- Finding the best model respecting the nature of data / queries
  - Not always possible

Current trends

- Multiple models within just a single system

Sample Database

Multi-model scenario

graph Friends

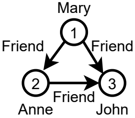


table Customer

CustomerId	Firstname	Lastname
1	Mary	Smith
2	Anne	Maxwell
3	John	Newlin

table Credit

CustomerId	Credit
1	30
2	25
3	30

column family Orders

CustomerId	Orders
1	[220, 230, ...]
2	[10, 217, ...]
3	[94, 137, 214]

collection Order

```
{ OrderID : 220,
  Items : [ {
    ProductID : B1,
    Name : Fairy Tales,
    Quantity : 1 } ] }
{ OrderID : 217,
  Items : [ {
    ProductID : T1,
    Name : Toy Car,
    Quantity : 2 } ] }
```

collection Product

```
{ ProductID : B1,
  Kind : Book,
  Name : Fairy Tales,
  Price : 20 }
{ ProductID : T1,
  Kind : Toy,
  Name : Toy Car,
  Price : 35 }
```

Existing Strategies

Polyglot persistence

- Different databases for different data models
  - Accessed independently or using an integrating mediator
- Academic proposals
  - E.g.: DBMS+, BigDAWG

Multi-model databases

- One database for multiple different data models
  - Provides a fully integrated backend
- More than 20 representatives
  - E.g.: OrientDB, ArangoDB, MarkLogic, Virtuoso, …

Multi-Model Databases

Issues and challenges

- **Underlying models**
  - Number of supported models, non-equal roles, …
- **Cross-model processing**
  - Links between the models, querying, indexing, …
- **Practical aspects**
  - Too many models and query languages
  - Data decomposition, specific features
  - Qualified users, deployment and maintenance
- **Formal background**
  - Proprietary solutions (often not well documented)

Paper Objectives

Formal **unifying framework** is necessary

- Solid theoretical background
- But still user-friendly enough

Our **objective**

- Draft of such a framework based on **category theory**
  - Conceptual modeling
  - Database decomposition
  - Data representation
  - Query evaluation
  - Evolution management

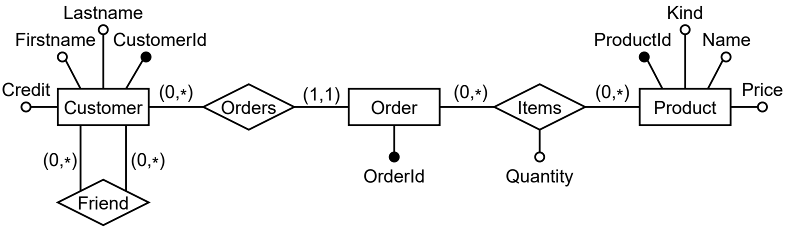
Category Theory

Category

- $\mathbf{C} = (\mathcal{O}, \mathcal{M}, \circ)$
  - Set of **objects** $\mathcal{O}$ (acting as multigraph vertices)
  - Set of **morphisms** $\mathcal{M}$ (acting as directed edges)
    - Each modeled as an arrow $f\colon A \to B$ with objects A, B
  - **Composition** operation $\circ$ for the morphisms
- Requirements
  - **Transitivity**: $g \circ f \in \mathcal{M}$ for any suitable morphisms f, g
  - **Associativity**: $h \circ (g \circ f) = (h \circ g) \circ f$ for any suitable f, g, h
  - **Identities**: identity morphism 1_A for any object A such that $f \circ 1_A = f = 1_B \circ f$ for any suitable morphism f
- Example
  - $\mathbf{Set}$: objects are sets, morphisms functions between them

Conceptual Modeling

Conceptual schema (ER)

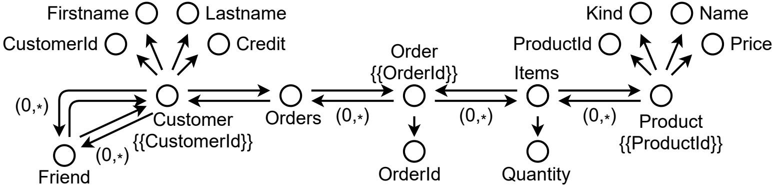


- Not standardized, various notations, structured attributes, identifiers for relationship types, participants of weak relationship types, non-unique or ordered values, …

Conceptual Modeling

Schema category

- **Objects** (attribute, entity, relationship)
  - Name, identifiers, superidentifier
- **Morphisms** (attribute, relationship, hierarchy)
  - Cardinality

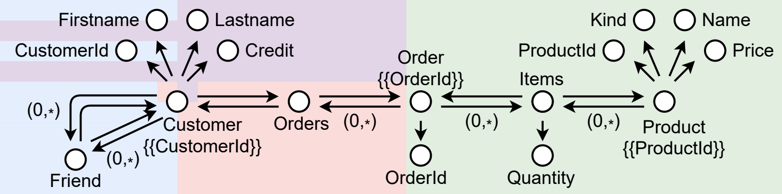


- Transformation from ER / design from scratch
- Higher expressive power

Database Decomposition

Mapping category

- Mapping of objects / morphisms to database components



- Components may overlap each other
  - Allows for intentional redundancies
- They may also be disconnected

Data Representation

Instance category

- Describes one particular database instance
- Objects and morphisms
  - Analogous to a given **schema category**
  - Internally contain sets of tuples

ProductId
B1
T1

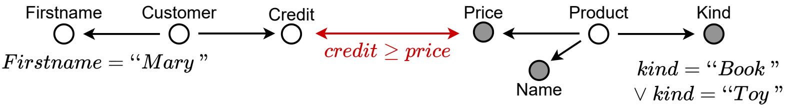
ProductId	Name
B1	Fairy Tales
T1	Toy Car

Name
Fairy Tales
Toy Car

Query Evaluation

Query category

- Based on **subgraph pattern matching**
- Objects and morphisms
  - Mapped to a given **schema category**
  - Filtering conditions, joining morphisms



- Query
  - Names, kinds, and prices of all books or toys which can be bought by a customer with name Mary

Query Evaluation

Query decomposition

- Based on schema category decomposition
  - Several plans may be possible
  - Cost estimation needed

- Individual query parts are then evaluated separately

Query Evaluation

Query translation

- **Relational** component
  - ```
    SELECT T2.Credit
    FROM Customer AS T1 NATURAL JOIN Credit AS T2
    WHERE T1.Firstname = "Mary";
    ```
- **Document** component (MongoDB)
  - ```
    db.Product.find(
      { Kind: { $in: [ "Book", "Toy" ] } },
      { ProductId: 0 }
    );
    ```

Query Evaluation

Result completion

- Intermediate results are transformed and combined

Credit
30

{ { Kind : Book,
Name : Fairy Tales,
Price : 20 },
{ Kind : Toy,
Name : Toy Car,
Price : 35 } }

Kind	Name	Price
Book	Fairy tales	20

Framework Features

Features and consequences

- **One homogeneous structure** for schema / data / queries
- Higher expressive power of **schema categories**
- Unified handling of **attributes, entity and relationship types**
- Merging of **conceptual and logical layers**
- **Non-awareness of decomposition** when querying
- Native support for **inter-model links** and cross-model querying
- …

Conclusion

Unifying framework for **multi-model systems** is necessary

- **Category theory** seems to be promising enough

Particular **challenges**

- **Conceptual modeling**
- **Schema inference**
- **Database decomposition**
- **Data representation**
- **Transformation operations**
- **Query evaluation**
- **Evolution management**
- **Autonomous tuning**

Thank you for your attention…