Institutions for database schemas and datasets

Martin Glauer[1] and Till Mossakowski[2]

[1]Otto von Guericke University, Magdeburg, Germany, glauer@iks.cs.ovgu.de
[2]Otto von Guericke University, Magdeburg, Germany, till@iks.cs.ovgu.de

1 Introduction

Database techniques have a long tradition in computer science and build the foundation for numerous modern applications. There are many category-theoretical approaches tackling several problems that occur when working with databases. Modelling database schemas as categories yields the intuitive notion of a schema merge as pushouts of functors [4]. A similar approach towards schema integrations was definedfor the first time in an institutional setting in [1]. The defined structures are not close to actual relational database structures. Yet, an institutional approach towards database structures yields functionalities not only on a structural, but also on the data level. Institutions were defined in [2] as a framework to cover the vast landscape of logical formalisms. The formalization presented in this paper can be a first step towards institution based logical reasoning on relational databases.

2 Formalization

We present an institution that, in contrast to [4], closely follows the Data Description Language (DDL) and the Data Manipulation Language (DML) of modern SQL-based database systems.

An object Σ of the **signature** category corresponds to a schema formulated in the DDL excluding constraint definitions. It consists of a sef of tables $\mathbb{T}^\Sigma$, a set of columns $\mathbb{C}^\Sigma$, a set of types $\mathbb{S}^\Sigma$, a family of sets of functional symbols $(\mathcal{F}_{\Sigma,w,s})_{w\in(\mathbb{S}^\Sigma)^*,s\in\mathbb{S}^\Sigma}$, and a family of sets of predicate symbols $(\mathcal{P}_{\Sigma,w})_{w\in(\mathbb{S}^\Sigma)^*}$. Additionaly, it contains functions that link tables and columns $\mathrm{col}^\Sigma(\cdot):\mathbb{T}^\Sigma\to\wp(\mathbb{C}^\Sigma)$ and columns to their types $\tau(\cdot,\cdot):\{(t,c)|t\in\mathbb{T}^\Sigma,c\in\mathrm{col}(t)\}\to\mathbb{S}^\Sigma$.
In the following we will use the abbreviation $\mathbb{C}^\Sigma\{\mathbb{T}\}:=\{(t,c)|t\in\mathbb{T}^\Sigma,c\in\mathrm{col}^\Sigma(t)\}$.

A **signature morphism** $\sigma:\Sigma\to\Sigma'$ consists of functions $\sigma_\mathbb{T}:\mathbb{T}^\Sigma\to\mathbb{T}^{\Sigma'}$, $\sigma_\mathbb{S}:\mathbb{S}^\Sigma\to\mathbb{S}^{\Sigma'}$. σ_{col} maps tables t to functions $(col(t)\to col'(\sigma_\mathbb{T}(t)))$ on their respecive column spaces, $\sigma_\mathcal{F}$ such that all n-ary function symbols $f\in\mathcal{F}_{\Sigma,w_1,\ldots,w_n,s}$ are mapped to $\sigma_\mathcal{F}(f)\in\mathcal{F}_{\Sigma,\sigma_\mathbb{S}(w_1),\ldots,\sigma_\mathbb{S}(w_n),\sigma_\mathbb{S}(s)}$ and $\sigma_\mathcal{P}$ respectively. Additionally, the types of columns must be preserved along $\sigma_\mathbb{S}$.

A **sentence** φ in $Sen(\Sigma)$ consists of a table $t\in\mathbb{T}^\Sigma$ and a constraint where the latter can can be one of the following: A primary key $pk\in\mathrm{col}(t)$, a foreign key $fk\in\mathrm{col}(t)\times\mathbb{C}\{\mathbb{T}\}$, a check constraint ck that is an unquantified first-order formula over the variables $\mathrm{col}(t)$ or a uniqueness constraint $un\in\mathrm{col}(t)$. The translation of these sentences along a signature morphisms is the intuitive application of the corresponding functions.

An object M of the **model** category $\mathrm{Mod}(\Sigma)$ of a signature Σ represents the data stored in each table as well as interpretation of the sorts named in Σ. Each model M consists of non-empty

carrier sets M_s for $s \in \mathbb{S}^\Sigma$, a function $(f_{w_1 \ldots w_n s})_M : M_{w_1} \times \ldots \times M_{w_n} \to M_s$ for each function symbol $f \in \mathcal{F}_{w_1,\ldots,w_n,s}$, a relation $(p_{w_1 \ldots w_n})_M \subseteq M_{w_1} \times \ldots \times M_{w_n}$ for each predicate symbol $f \in \mathcal{P}_{w_1,\ldots,w_n}$, a function *data* that maps each table $t \in \mathbb{T}^\Sigma$ to a (multi-)set of functions $(\mathrm{col}(t) \to M_{\tau(t,c)})$.

The **reduct** $M'|_\sigma$ of a model M' in $\mathrm{Mod}(\Sigma')$ against a signature morphism $\sigma : \Sigma \to \Sigma'$ consists of carrier sets $(M'|_\sigma)_s = M'_{\sigma_\mathbb{S}(s)}$. Analogously, function and predicates are obtained by translating their sorts along $\sigma_\mathbb{S}$. The data state of a table depends on the images of its columns:

$$\forall t \in \mathbb{T}^\Sigma, c \in \mathrm{col}(t) : data_{M|_\sigma}(t) = data_M(\sigma_\mathbb{T}(t)) \circ (\sigma_{\mathrm{col}}(t)) \tag{1}$$

For every $M \in \mathrm{Mod}(\Sigma)$ and $\varphi \in Sen(\Sigma)$ the **satisfaction relation** $M \models_\Sigma \varphi$ holds depending on the structure of φ: If $\varphi = (t, c)$ is a uniqueness or a primary key constraint:

$$\forall row, row' \in data_M(t) : \big(row \neq row' \Rightarrow row(c) \neq row(c')\big)$$

If $\varphi = (t, (c, (t', c')))$ is a foreign key constraint:

$$\forall row \in data_M(t), \exists row' \in data_M(t') : row(c) = row'(c')$$

If $\varphi = (t, ck)$ is a check key constraint it is evaluated as a first order formula:

$$\forall row \in data_M(t) : row \models_\Sigma{}^{FOL} ck$$

Whilst the models in the database institution described above represent a possible data state, morphisms of the category of models are the transitions between these states, i.e. statements of the DML like *INSERT*, *UPDATE*, *DELETE*. Similar morphism structures have been used in a categorical approach towards version control systems [3].

Consider a data state M and two different, concurrent manipulations (i.e. morphisms) $m_1 : M \to M_1$, $m_2 : M \to M_1$. If there is a pushout $m'_1 : M_1 \to M^*$, $m'_2 : M_2 \to M^*$ it is possible to merge both changes directly into a single data state M^*.

3 Conclusion

Whilst current approaches focus mainly on the schematic transformations of databases, the presented institution also covers the behavior on the data level. This allows the definition of formal foundations for collaborative database systems. These foundations can be used to evaluate existing collaborative database systems with respect to formal correctness or develop new such systems.

References

[1] Suad Alagić and Philip A Bernstein. A model theory for generic schema management. In *International Workshop on Database Programming Languages*, pages 228–246. Springer, 2001.

[2] Joseph A Goguen and Rod M Burstall. Institutions: Abstract model theory for specification and programming. *Journal of the ACM (JACM)*, 39(1):95–146, 1992.

[3] Samuel Mimram and Cinzia Di Giusto. A categorical theory of patches. *Electronic notes in theoretical computer science*, 298:283–307, 2013.

[4] Patrick Schultz, David I Spivak, Christina Vasilakopoulou, and Ryan Wisnesky. Algebraic databases. *arXiv preprint arXiv:1602.03501*, 2016.