Theory I: Database Foundations

Jan-Georg Smaus (Georg Lausen)

20.7.2011

1. Introduction
   Intuition
   Formalization
Introduction

Consider a student database.

The represented information:

- Hans Eifrig is assigned Matrikelnummer 1223. His address is Seeweg 20. He is studying in the second semester.
- Lisa Lustig is assigned Matrikelnummer 3434. Her address is Bergstraße 11. She is studying in the fourth semester.
- Maria Gut is assigned Matrikelnummer 1234. Her address is Am Bächle 1. She is studying in the second semester.
Relational databases use tables to represent information

### Student

<table>
<thead>
<tr>
<th>MatrId</th>
<th>Name</th>
<th>Address</th>
<th>Semester</th>
</tr>
</thead>
<tbody>
<tr>
<td>1223</td>
<td>Hans Eifrig</td>
<td>Seeweg 20</td>
<td>2</td>
</tr>
<tr>
<td>3434</td>
<td>Lisa Lustig</td>
<td>Bergstraße 11</td>
<td>4</td>
</tr>
<tr>
<td>1234</td>
<td>Maria Gut</td>
<td>Am Bächle 1</td>
<td>2</td>
</tr>
</tbody>
</table>

### Course

<table>
<thead>
<tr>
<th>CourseId</th>
<th>Institute</th>
<th>Title</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>K010</td>
<td>DBIS</td>
<td>Databases</td>
<td>Foundations of Databases</td>
</tr>
<tr>
<td>K011</td>
<td>DBIS</td>
<td>Information Systems</td>
<td>Foundations of Information Systems</td>
</tr>
</tbody>
</table>

### Registration

<table>
<thead>
<tr>
<th>MatrId</th>
<th>CourseId</th>
<th>Semester</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>1223</td>
<td>K010</td>
<td>WS2017/2018</td>
<td>2.3</td>
</tr>
<tr>
<td>1234</td>
<td>K010</td>
<td>SS2018</td>
<td>1.0</td>
</tr>
</tbody>
</table>
Objects and relationships

- An “object” in a database is a **tuple**.
- Each argument of the tuple represents the **value** of some **attributes**.
- Some of the attributes are called **keys**. Objects can be distinguished by their **key values**.
- A set of tuples is a **relation**.
- One or more relations constitute a **database**.
Using a database

- Application programs communicate with a database to query, update, insert and delete the state of the database.
- All these operations use some query language, say SQL.
- Query expressions have a set-oriented, declarative semantics:
  - The result of a query is a set of tuples.
  - The query defines the what and not the algorithmically how.
- Given the what, an optimizer can try to improve the efficiency of the query evaluation.
What are the names of the 'DBIS'-professors?

SELECT P.Name
FROM Professor P
WHERE P.Institute = 'DBIS'

Which students are registered for which courses?

SELECT S.Name, K.Title
FROM Student S, Registration B, Course K
WHERE S.MatrId = B.MatrId AND
B.CourseId = K.CourseId
We note . . .

- A relational database - or simply database - uses relations (tables) to represent the information required for a certain business, i.e. tasks of an enterprise, web portal, or even your personal life.
- We also say: a database represents a relevant state of its environment.
- We distinguish the definition of the structure – the type – of a relation from its concrete time-dependant state – the value.
- The schema of a relation refers to the type, the instance to a certain value, i.e. a set of tuples, respectively rows, if we think of a table.
We start with attributes

- A tuple ("object") is identified by its "properties", which we call attributes.
- Let \( X = \{A_1, \ldots, A_k\} \) be a (finite) set of attributes, \( k \geq 1 \).
- Each attribute \( A \in X \) is assigned a non-empty domain \( \text{dom}(A) \).
- \( \text{dom}(X) = \bigcup_{A \in X} \text{dom}(A) \).

Example

The attribute Colour may have the domain \( \{\text{red, green, \ldots}\} \).
The attribute Semester may have the domain \( \{1, 2, \ldots\} \) (what do you think is reasonable?).
Tuple

- Attribute values (one value for each attribute) can be grouped to form a tuple.
- Formally, a tuple \( \mu \) over \( X \) is a mapping

\[
\mu : X \rightarrow \text{dom}(X),
\]

where \((\forall A \in X)\mu(A) \in \text{dom}(A)\).
- \( \text{Tup}(X) \) is defined as the set of all tuples over \( X \).

Example

\( \mu_1 = \{ \text{MatrId} \rightarrow 1223, \text{Name} \rightarrow \text{Hans Eifrig}, \text{Address} \rightarrow \text{Seeweg 20}, \text{Semester} \rightarrow 2 \} \)

\( \mu_2 = \{ \text{MatrId} \rightarrow 3434, \text{Name} \rightarrow \text{Lisa Lustig}, \text{Address} \rightarrow \text{Bergstraße 11}, \text{Semester} \rightarrow 4 \} \)

\( \mu_3 = \{ \text{MatrId} \rightarrow 1234, \text{Name} \rightarrow \text{Maria Gut}, \text{Address} \rightarrow \text{Am Bächle 1}, \text{Semester} \rightarrow 2 \} \)
Tuples: mappings vs. vectors

\[ \mu_1 = \{ \text{MatrId} \rightarrow 1223, \text{Name} \rightarrow \text{Hans Eifrig, } \]
\[ \text{Address} \rightarrow \text{Seeweg 20, Semester} \rightarrow 2 \} \]

\[ \mu' = \{ \text{MatrId} \rightarrow 1223, \text{Address} \rightarrow \text{Seeweg 20, } \]
\[ \text{Semester} \rightarrow 2, \text{Name} \rightarrow \text{Hans Eifrig} \} \]

(1223, Hans Eifrig, Seeweg 20, 2)
(1223, Seeweg 20, 2, Hans Eifrig)

What kind of equality would you expect for tuples?
A relation $r$ over $X$ is a finite set $r \subseteq \text{Tup}(X)$.

The set of all relations over $X$ is denoted $\text{Rel}(X)$.

$r \in \text{Rel}(X)$ is called an instance over $X$.

Let $R$ be a relation name. A (relation) schema of $R$ is given as $R(X)$, where $X$ a set of attributes, also called format of the schema.

Instead of writing $R(\{A_1, \ldots, A_k\})$ we may also write $R(A_1, \ldots, A_k)$. $k$ is called the arity of $R$.

We may also write:

$$R(A_1 : \text{dom}(A_1), \ldots, A_k : \text{dom}(A_k))$$
### Key

- For each schema $R(X)$ we distinguish a set of attributes $K$ we call a **key** of $R$, $K \subseteq X$.

- Once a key is defined, in every instance $r$ of $R$ for every pair of tuples it holds that: if both tuples agree on the attributes forming the key, they have to agree on all their attributes.

- In general, for a schema there may exist several keys.
A (relational) database schema $\mathcal{R}$ is given as a set of relation schemata,

$$\mathcal{R} = \{R_1(X_1), \ldots, R_m(X_m)\},$$

resp. $\mathcal{R} = \{R_1, \ldots, R_m\}$.

An instance $\mathcal{I}$ of a database schema $\mathcal{R} = \{R_1, \ldots, R_m\}$ is given as a set of finite relations, $\mathcal{I} = \{r_1, \ldots, r_m\}$, where $r_i$ instance of $R_i$, $1 \leq i \leq m$. We may also write

$$\mathcal{I}(R_i) = r_i, 1 \leq i \leq m.$$
For any instance $\mathcal{I}$, a query $Q$ defines a relation $Q(\mathcal{I})$, we call the answer to $Q$.

A query is formally given as a mapping (transformation) from a database instance to a relation instance.

Analogously to above, we may also write $\mathcal{I}(Q)$ to denote the answer to a query $Q$ with respect to an instance $\mathcal{I}$. 
Null value

- We may introduce a null value, whenever we want to express, that for some attribute the value is not known.

- The problem with nulls is that there exist several different possible interpretations: value exists, however currently not known; value currently does not exist, however will exist in the future; value exists, however is unknown in principle; and attribute is not applicable.

Example

<table>
<thead>
<tr>
<th>MatrId</th>
<th>Name</th>
<th>Address</th>
<th>Semester</th>
</tr>
</thead>
<tbody>
<tr>
<td>1223</td>
<td>Hans Eifrig</td>
<td>null</td>
<td>2</td>
</tr>
<tr>
<td>3434</td>
<td>Lisa Lustig</td>
<td>Bergstraße 11</td>
<td>4</td>
</tr>
<tr>
<td>1234</td>
<td>Maria Gut</td>
<td>Am Bächle 1</td>
<td>null</td>
</tr>
</tbody>
</table>