Measures for Schema Quality

Hans J. Lenz, Freie Universität Berlin

September 2007
1 Workflow: From Kick-off to DB System

Demand for mapping reality into a „mini world“.

Conceptual Design und Visualisation
(conceptual Data Model)

Normalised Relational Model
(logical Data Model)

Implementing DB System

Schema Quality
### Illustration: Modelling Variants

#### Person

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Surname</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John</td>
<td>Smith</td>
<td>113 Sunset Avenue 60601 Chicago</td>
</tr>
<tr>
<td>2</td>
<td>Mark</td>
<td>Bauer</td>
<td>113 Sunset Avenue 60601 Chicago</td>
</tr>
<tr>
<td>3</td>
<td>Ann</td>
<td>Swenson</td>
<td>4 Heroes Street Denver</td>
</tr>
</tbody>
</table>

#### Address

<table>
<thead>
<tr>
<th>ID</th>
<th>StreetPrefix</th>
<th>StreetName</th>
<th>Number</th>
<th>City</th>
</tr>
</thead>
<tbody>
<tr>
<td>A11</td>
<td>Avenue</td>
<td>Sunset</td>
<td>113</td>
<td>Chicago</td>
</tr>
<tr>
<td>A12</td>
<td>Street</td>
<td>4 Heroes</td>
<td>null</td>
<td>Denver</td>
</tr>
</tbody>
</table>

#### ResidenceAddress

<table>
<thead>
<tr>
<th>PersonID</th>
<th>AddressID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A11</td>
</tr>
<tr>
<td>2</td>
<td>A11</td>
</tr>
<tr>
<td>3</td>
<td>A12</td>
</tr>
</tbody>
</table>

Problem of Modelling (a):
- ambiguous values
- redundant data
- superfluous info

Problem of Modelling (b):
- complex Structure

trade-off
3 Seven Dimensions of Schema Quality

1. **Readability**
2. Normalisation
3. Correctness w.r.t. Model
4. Correctness w.r.t. Requirements
5. Minimalisation
6. Completeness
7. Pertinence („over modelling“)

Source: Redman (1996)
3.1 Readability of ERM / UML

DEF.: A schema is readable whenever it represents the meaning in the reality represented by the schema in a clear way for its intended use.

Aesthetic Criteria
- Avoid Crossing between arcs (prefer planar graph)
- Embed symbols in a grid
- Horizontal or vertical drawings of lines mandatory
- Minimum number of bends of lines
- Minimum Area of Diagram (one glimpse capturing)

Structural Adequacy
- Hierarchical Representations of Objects
- Symmetry of Children-Objects w.r.t. Parent-Objects

3.1 Readability of ERM

„Spaghetti“-Style:

Equivalent readable Schema:

3.1 Readability of ERM (2)

Two equivalent models showing is-a generalisation.

Compactness of (b) due to inheritance.

Seven Dimensions of Schema Quality

1. Readability ✓
2. Normalisation
3. Correctness w.r.t. Model
4. Correctness w.r.t. Requirements
5. Minimalisation
6. Completeness
7. Pertinence (over modelling)
3.2 Normalisation

DEF.: Loss-less Decomposition of a relational model (set of tables) in order to avoid redundancy and anomalies of data management.

Modelling:
- intuitive / rules of thumb / logical criteria
- identification of structural weakness
- informal (heuristic) or formal (Normalisation) criteria of correct relational designs
### Informal Criteria of Modelling

**Ex.: Structural deficits of a schema:**

<table>
<thead>
<tr>
<th>M-Nr</th>
<th>M-Name</th>
<th>M-GebDat</th>
<th>A-Nr</th>
<th>A-Bez</th>
<th>A-Leiter</th>
</tr>
</thead>
<tbody>
<tr>
<td>234</td>
<td>Müller</td>
<td>1.10.1959</td>
<td>1</td>
<td>Einkauf</td>
<td>234 376</td>
</tr>
<tr>
<td>345</td>
<td>Meier</td>
<td>20.01.1964</td>
<td>2</td>
<td>Marketing</td>
<td>345</td>
</tr>
<tr>
<td>376</td>
<td>Schmidt</td>
<td>15.06.1968</td>
<td>1</td>
<td>Einkauf</td>
<td>234</td>
</tr>
<tr>
<td>345</td>
<td>Schule</td>
<td>30.12.1985</td>
<td>9</td>
<td>Marketing</td>
<td>345</td>
</tr>
<tr>
<td>&lt;NULL&gt;</td>
<td>&lt;NULL&gt;</td>
<td>&lt;NULL&gt;</td>
<td>3</td>
<td>Produktion</td>
<td>&lt;NULL&gt;</td>
</tr>
</tbody>
</table>

**Insert-Anomaly** (Entity Integrity Constraint):
Null values not allowed!

**Delete-Anomaly:**
Loss of Information about facts

**Update-Anomaly:**
Inconsistencies if changes are not effective across full database.
3.2.2 Formal Criteria of Modelling

Normalisation
Steps of sequenced decomposition of relation types into subtypes

Normal Form
State of a Relation type Represents quality of design
3.2.3 Hierarchy of Normal Forms

Normal Forms are stacked

..Relations of level k satisfy restrictions of level h < k = 1, 2,...,5

1 NF: not normalized

Based on functional dependencies

Basic Prerequisite: atomic domains
3.2.3 Normal Forms

- **DEF.: 1. Normal Form (1NF)**
  All Attribute values of a schema must have *atomic* data types, i.e. sets, bags, arrays, records, lists, tables etc. not allowed

**Ex.: 0-NF**

<table>
<thead>
<tr>
<th>Nr</th>
<th>Name</th>
<th>Adresse</th>
</tr>
</thead>
<tbody>
<tr>
<td>234</td>
<td>Müller, Hans</td>
<td>Bismarkstr. 11, 10961 Berlin</td>
</tr>
<tr>
<td>345</td>
<td>Meier, Otto</td>
<td>Hüttenweg 32, 10944 Berlin</td>
</tr>
<tr>
<td>376</td>
<td>Schmidt, Jan</td>
<td>Bergmannstr. 25, 10174 Berlin</td>
</tr>
</tbody>
</table>

... Attribute *Name* is a concatenated string of family and first name

... Attribute *Adress* is a record
DEF.: Normalisation
map a set-valued attribute into a set of single-valued attributes

Poor Quality Solution:
use single attributes for each item. Note that the group assignment is lost

Good Quality Solution:
Define a separate table schema and link it to the original table by a foreign key - primary key relationship.
DEF.: Functional Dependency (FD)
Attribute B is functional dependent on attribute A, if for each value of A there exists only a unique value of B (true for groups of attributes, too).

Ex.:

<table>
<thead>
<tr>
<th>Teamwork</th>
</tr>
</thead>
<tbody>
<tr>
<td>Emp#</td>
</tr>
</tbody>
</table>

- Full FD
- Partial FD
- Partial FD
DEF.: 2nd Normal Form (2NF)
Table Schema is in 1NF and each non-key attribute must be fully dependent on each candidate key.

1NF: Teamwork

2NF: Teamwork

1NF but not 2NF

1NF and 2NF
DEFF.: Transitive Dependency

Attribute C is transitive dependent on candidate key A, if a non-key attribute B exists on which C is functional dependent where B itself is functional dependent on A.

Ex.: Employee

```
Emp#  Emp-Name  Pro#  Pro-Name
```

Diagram:

- Arrows indicating dependencies between attributes.
3.2.3 Normal Forms (6)

- **DEF.: 3rd Normal Form (3NF)**
  Table schema is 2NF and no non-key attribute is transitive dependent on any candidate key.

**2NF:**
- Employee
  - Emp#
  - Emp-Name
  - Pro#

**3NF:**
- Employee
  - Emp#
  - Emp-Name
  - Pro#

- **Project**
  - Pro#
  - Pro-Name

- **Diagram:**
  - In 2NF but not 3NF
  - In 2NF and 3. NF
DEF.: Determinant

Attribute A is a determinant if there exists at least another attribute B which is fully dependent of A.

Ex.:
3.2.3 Normal Forms (8)

- Boyce-Codd-Normalform (BCNF)
  A table schema is in Boyce-Codd Normal Form if each determinant is a candidate key.

```plaintext
before: club#  Back#  Phys#

after: Phys#  Back#
```

In 3NF but not in BCNF

in BCNF
3 Seven Dimensions of Schema-Qualität

1. Readability ✓
2. Normalisation ✓
3. Correctness w.r.t. model
4. Correctness w.r.t. requirements
5. Minimalisation
6. Completeness
7. Pertinence (over modelling)
3.3 Correctness w.r.t. Model

- **DEF.:** Correct Modelling as far as requirements are concerned

**Table:**

<table>
<thead>
<tr>
<th>Per#</th>
<th>First Name</th>
<th>Family Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Klaus</td>
<td>Meier</td>
</tr>
<tr>
<td>2</td>
<td>Hans</td>
<td>Müller</td>
</tr>
<tr>
<td>3</td>
<td>Otto</td>
<td>Schmidt</td>
</tr>
</tbody>
</table>

**Diagrams:**

- **Not correct:** Object *FirstName* is not a real object but a category.
- **Correct:** *FirstName* as Attribute in schema *Person*.
3.4 Correctness w.r.t. Requirements

DEF.: Correctness w.r.t. to requirements is the correct representation of constraints / requirements in terms of object categories

Business Rule:
Each department is headed by exactly one manager and each manager is the head of exactly one department.

3.3-3.4: Mis-Specification of a cube
The Cattle-Example of Snodgrass

- Snodgrass (1999) defines a data cube (4-way contingency table) "Count of cattle grouped by lot, pen and date"
- The categorical attribute (dimension) 'date' is split into the two sub-attributes 'from_date' and 'to_date'.
- Fact Table:

<table>
<thead>
<tr>
<th>count</th>
<th>lot-id</th>
<th>pen-id</th>
<th>from-data</th>
<th>to-data</th>
</tr>
</thead>
</table>

Source: Sno99, chap. 11 "Conceptual Design".
ERM of "Cattle"-Example
(revised version on OLTP level)

Source: Lenz and Thalheim (2002)
Relational Modelling

Model by Snodgrass (1999)

FDYD (Fdyd_ID, Name,…)
LOT(Fdyd_ID, Lot_ID-Num, Lot_Id, Gndr_Code,…)
Pen(Fdyd_ID, Pen_ID,Pen_Type_Code,…)
Application (A_Name, A_Description,…)
DBF_File(A_Name, DBF_Name,…)
BKP(Fdyd_ID, BKP_Id, …)

Model by Lenz and Thalheim (2002)

Cattle (Cattle_ID, BelongsTo, …)
Lot (Lot_ID, …)
Resides (Cattle_ID, Pen_ID, From, To, …)
Pen (Pen_ID, …)
Query: "Find the History of Lots being co-resident in a Pen"

```sql
select L1.Lot_Id_num, L2.Lot_Id_Num, L1.Pen_Id, L1.From_Date, L1.To_Date
from Lot_Loc as L1, Lot_Loc as L2
where L1.Lot_Id_num < L2.Lot_Id_num
    and L1.Fdyd_Id = L2.Fdyd_Id and L1.Pen_Id = L2.Pen_Id
    and L1.From_Date = L2.From_Date and L1.To_Date <= L2.To_Date

union

select L1.Lot_Id_num, L2.Lot_Id_Num, L1.Pen_Id, L1.From_Date, L2.To_Date
from Lot_Loc as L1, Lot_Loc as L2
where L1.Lot_Id_num < L2.Lot_Id_num
    and L1.Fdyd_Id = L2.Fdyd_Id and ...

union

select L1.Lot_Id_num, L2.Lot_Id_Num, L1.Pen_Id, L2.From_Date, L1.To_Date
from Lot_Loc as L1, Lot_Loc as L2
where L1.Lot_Id_num < L2.Lot_Id_num
    and L1.Fdyd_Id = L2.Fdyd_Id and ...

union

select L1.Lot_Id_num, L2.Lot_Id_Num, L1.Pen_Id, L2.From_Date, L2.To_Date
from Lot_Loc as L1, Lot_Loc as L2
where L1.Lot_Id_num < L2.Lot_Id_num
    and L1.Fdyd_Id = L2.Fdyd_Id and L1.Pen_Id = L2.Pen_Id
    and L1.From_Date > L1.From_Date and L2.To_Date <= L1.To_Date;
```
Query: "Find the History of Lots being co-resident in a Pen"

select distinct L1.Lot_ID, L2.Lot_ID, R1.Pen_ID, R2.From, min(R1.To, R2.To)
from Cattle C1, Cattle C2, Resides R1, Resides R2, Lot L1, Lot L2
where L1.Lot_ID = C1.BelongsTo and L2.Lot_ID = C2.BelongsTo and
   R1.Cattle_ID = C1.Cattle_ID and
   R2.Cattle_ID = C2.Cattle_ID and
   R1.Pen_ID = R2.Pen_ID and
   R1.From <= R2.From and R2.From < R1.To and L1.Lot_ID <> L2.Lot_ID.
3.5 Minimalisation

DEF.: A Schema is minimal if each part of the requirements is represented only once.

- Relational Type \textit{enrolled} is redundant iff upper bound of cardinality \( ? = 1 \).
3.6 Completeness

- DEF.: Extent to which a schema includes all objects necessary to meet some specified conceptual requirements

Ex.:

<table>
<thead>
<tr>
<th>Per#</th>
<th>Firstname</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Klaus</td>
<td>Seestr. 2</td>
</tr>
<tr>
<td>2</td>
<td>Hans</td>
<td>Garystr. 12</td>
</tr>
<tr>
<td>3</td>
<td>Otto</td>
<td>Heerstr. 10</td>
</tr>
</tbody>
</table>

Relation Person is not complete because attribute FamilyName is missing.

3.6 Completeness (cont.)

ERD for Metadata

Source: Lenz(1994)
3.6 Completeness (cont.)
ERD for Metadata of entity „Attribute“

Source: Lenz (1994)
3.6 Completeness (cont.)  Metadata

Metadata describe universes (populations), micro and macro data on the levels

- *semantic*,
- *structural*,
- *statistical*, and
- *physical*

in such a way that

- the universe is well defined, and data can be reasonably
- inputted, stored, updated,
- transformed, grouped, summarized (aggregated),
- retrieved and disseminated.
3.7 Pertinence („over modelling“)

- Number of unnecessary objects included in the schema

- Ex:

<table>
<thead>
<tr>
<th>Per#</th>
<th>Firstname</th>
<th>FullName</th>
<th>Address</th>
<th>Hair Colour</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Klaus</td>
<td>Meier</td>
<td>Seestr. 2</td>
<td>brown</td>
</tr>
<tr>
<td>2</td>
<td>Hans</td>
<td>Müller</td>
<td>Garystr. 12</td>
<td>black</td>
</tr>
<tr>
<td>3</td>
<td>Otto</td>
<td>Schmidt</td>
<td>Heerstr. 10</td>
<td>blond</td>
</tr>
</tbody>
</table>

„Over Modelling“: „Hair colour“ is unnecessary for a citizen register
Note: Eye colour may be needed!
End of Schema Quality Dimensions

Good enough is not „good enough“!
4. Literatur


2. Dombrowski, Erik und Lechtenböger, Jens: Evaluation objektorientierter Ansätze zur Data-Warehouse-Modellierung, Datenbank-Spektrum 15/2005