Query and Answer Forms for Sophisticated Database Querying

Sophisticated NoSQL Questioning of a Database in Native Form

DBDBD'2013

Dutch-Belgian Database Day, Rotterdam, World-Trade Center

29.11.2013

Prof. Dr. Bernhard Thalheim

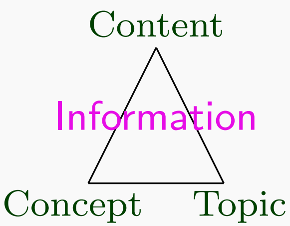
Technologie der Informationssysteme

Institut für Informatik, Christian-Albrechts-Universität zu Kiel, BRD

Kolmogorow-Professor e.h. der Lomonossow-Universität Moskau

What you would expect from this talk and what I do not deliver

- Open problems of information systems research and technology; BIR 2013, LNBIB 158, 2013.
- Conceptual modelling theory and practice; Handbook of conceptual modelling, 2011 or LNCS 7260, 6520, ...
- Constraints, database semantics; SDKB, LNCS 7693 or FoIKS 2012, LNCS 7153.
- The conception of the model; BIS 2013 or EJC 2013 or ... or ... .
- Entity relationship modelling; HERM book.
- Big data; ???
- Evolution and migration of information systems; Handbook of conceptual modelling, 2011 or DKE 87.
- Foundations of BPMN and workflows;
- Service theory; JUCS 18, ...



Plan for this Talk

Observations for the current state-of-art

- Trapped by SQL and database schemata
- Being limited for formulation, understanding, culture

Systematic querying by reconsidering search

- Property-based search is the toughest form of search!

Query forms as a framed form for query formulation

- Questions are anyway stereotyped.
- Use the stereotype for query generation.

Answer forms as a way of deriving the format of the answer

- Questions contain partially the answer format.
- Use the answer format for answer stereotypes.

Query formulation from questions

- SQL users have to state queries in the SQL form!

Question liquefaction for generation of queries

- Automatic query decomposition, liquefaction and composition.
- Natural language approaches to generation.

Weaknesses of SQL

NoSQL did not only appear because of big data ...

- Equivalent queries may produce different results.
- Aggregation operators like `SUM`, `AVG`, etc. doesn't generate the correct calculation in certain cases.
- Query tables that have nulls may lead us to misinterpret results in a variety of cases.
- Surface level coding ("The data is the code" and wrong injections) instead of source level coding: better we use parameterised expressions.
- Complex becomes more complex than it should be.
- Database query development is a matter of the skilled programmer.
- SQL does not have its visualisation.
- Users do not speak the 'intergalactic database speak'!

Not Yet Convinced on the Power of Visual Reasoning

Lets look onto Visual Literacy:

http://www.visual-literacy.org/periodic_table/periodic_table.html?goback=.gmp_2244682.gde_2244682-_member_5798121348945432578#!

There you will find tools, tools, ... for data visualisation, information visualisation, concept visualisation, strategy visualisation, metaphor visualisation and compound visualisation.

Visual reasoning for constraints: functional dependencies, multivalued dependencies, inclusion dependencies
simple and easy to understand and to develop.

Tufte principles for displays, visualisation of data: show the data, tell the truth, help the viewer think about the information, rather than to think about the design, encourage the eye to compare the data, make large data sets coherent.

State-of-art
Systematics
VisualSQL
Query forms
Answer forms
Question2Query
Liquefaction
Conclusion

Content
Information
Concept
Topic

SQL Query Generation

Static query interfaces

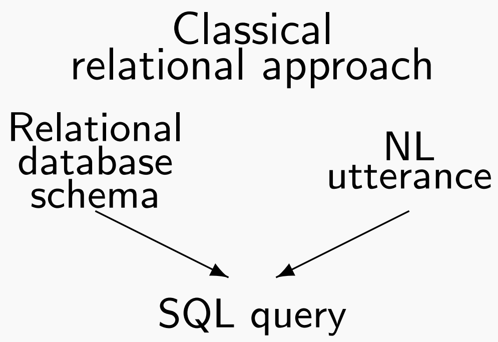
No or very restricted NL access

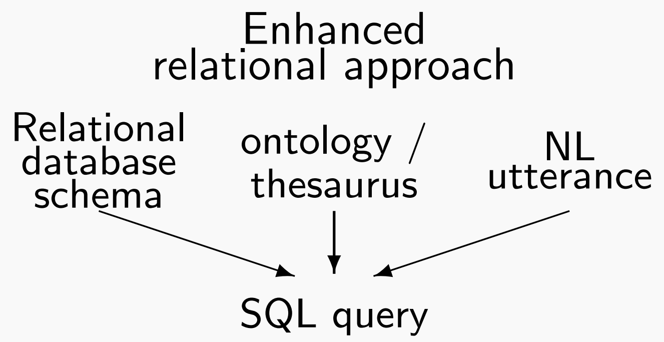
Simplicity of query interfaces

Problematic IR solutions

ER querying is better than relational but not widely used

Problematic database evolution





SQL is Easy to Read, to Develop and to Understand? Of Course, for Everybody!!!

What is the results of this query?

```
SELECT P1.Name, P2.Name
   FROM Person P1, Person P2, Student S1, Student S2, Enrol H1, Enrol H2
   WHERE P1.Name = S1.Name AND P1.DateOfBirth = S1.DateOfBirth AND
       S1.StudNo = H1.StudNo AND H1.Grade IS NOT NULL AND
       P2.Name = S2.Name AND P2.DateOfBirth = S2.DateOfBirth AND
       S2.StudNo = H2.StudNo AND H2.Grade IS NOT NULL
       AND S1.StudNo < S2.StudNo AND
       NOT EXISTS (
           SELECT * FROM Vorlesung AS V
               WHERE V.CourseNo IN
               (SELECT B.CourseNo FROM Enrol AS B
                   WHERE S1.StudNo = B.StudNo
                   OR S2.StudNo = B.StudNo)
               AND
               NOT EXISTS (
                   ( SELECT * FROM Enrol AS B1
                   WHERE S1.StudNo = B1.StudNo
                       AND B1.CourseNo = V.CourseNo )
                       UNION
                   ( SELECT * FROM Enrol AS B2
                   WHERE S2.StudNo = B2.StudNo
                   AND B2.CourseNo = V.CourseNo )
                   )
               )
   GROUP BY P1.Name, P2.Name;

;
```

Person, Student, Lecture, Enroll

SQL is Easy to Read, to Develop and to Understand? Of Course, for Everybody!!!

What does this query? What is the difference to the previous query?

```
SELECT P1.Name, P2.Name
   FROM Person P1, Person P2, Student S1, Student S2, Enrol H1, Enrol H2
   WHERE P1.Name = S1.Name AND P1.DateOfBirth = S1.DateOfBirth AND
       S1.StudNo = H1.StudNo AND H1.Grade IS NOT NULL AND
       P2.Name = S2.Name AND P2.DateOfBirth = S2.DateOfBirth AND
       S2.StudNo = H2.StudNo AND H2.Grade IS NOT NULL
       AND NOT EXISTS
       (SELECT *
           FROM Enrol H3
           WHERE H3.Grade IS NOT NULL AND
               H3.StudNo NOT IN
               (SELECT H4.StudNo
                   FROM Enrol H4
                   WHERE H4.StudNo = H2.StudNo
                       AND H4.Grade IS NOT NULL)
                       AND H1.StudNo = H3.StudNo)
       AND NOT EXISTS
       (SELECT *
           FROM Enrol H5
           WHERE H5.Grade IS NOT NULL AND
               H5.StudNo NOT IN
               (SELECT H6.StudNo
                   FROM Enrol H6
                   WHERE H6.StudNo = H1.StudNo
                       AND H4.Grade IS NOT NULL)
                       AND H2.StudNo = H5.StudNo)
       AND S1.StudNo < S2.StudNo
   GROUP BY P1.Name, P2.Name;
```

Person, Student, Lecture, Enrol

Peculiarities of the State-of-the-Art

- SQL, Codasyl are multi-set-based languages, relations are sets
- SQL is often taught through the tuple calculus
- SQL means only querying;
  computation, integrity maintenance, indexes, ... is for the artisan;
  triggers and stored procedures are inventions of the devil
- There is not **the** SQL; instead we have PL SQL, Transact SQL, ...
- Standards development is protection for big business
- Cookbooks, cookbooks, cookbooks for syntax
- Very few systematic and didactic books, almost no tricks
  We (CAU@Kiel) are teaching "advanced database programming"!!
- SQL processing is concurrent and context-sensitive processing

Myths of SQL Books

No tree and graph computation: yes if you switch off your brain

Recursion is not representable: is there one recursion? is it not representable?

Aggregation comes for free: independent of data sets, independent on their properties, attributes do not change their meaning in queries

No conceptual tuning and performance improvement: only logical or physical tuning

SQL is guilty for bad design: structure optimisation is still based on normalisation for machines of the 80ies
the non-sense of overloaded values and markers, e.g. NULL

Pitfalls of Computer Engineering Education

Logicians and discrete mathematicians were the first founders; therefore first-order predicate calculus is the only way of reasoning support

Programming could be thinking first but ...

Linear, sequential behaviour as a must: the world is however concurrent, parallel, coopetiting

Humans have to learn the way the programmer reasons: learn the system, read the manual - if it is coherent and consistent, don't ask

Hypes, hypes, hypes because of missing culture

Triptych programming in the Java age: program + library + exceptions
structural programming is old stuff that we have overcome

We are Humans!!

and thus we are

limited in our formulation capabilities what hampers the user,

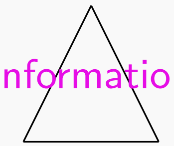
limited in abstraction skills what limits jumping into somebodies context,

not keen to learn database schemata which have been created by somebody whom we do not know or understand,

not able to read exhaustive result sets what requires sophisticated presentation, visualisation and compactification,

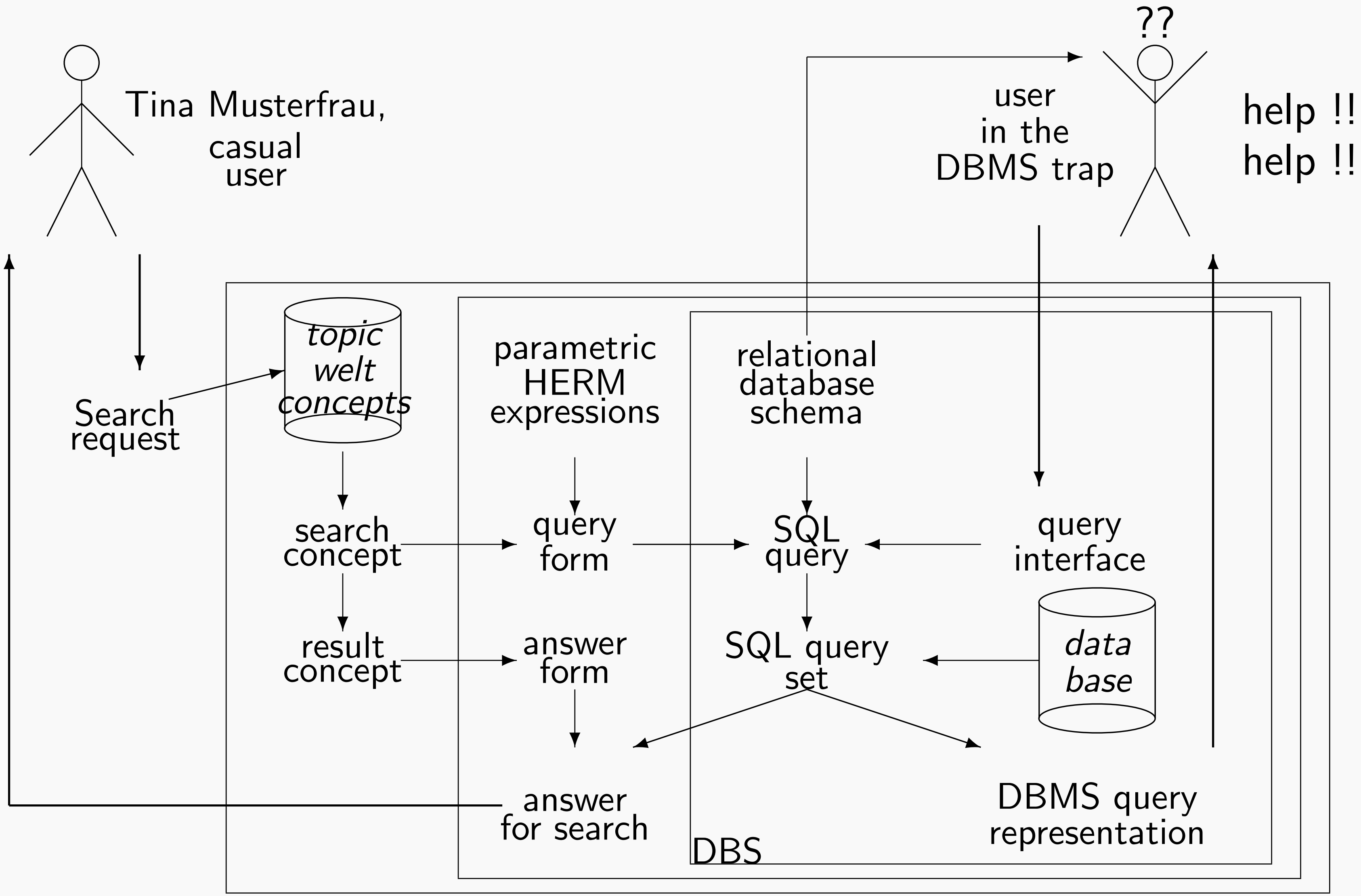
finite and bounded what means that we need support for parsimony of our memory, and

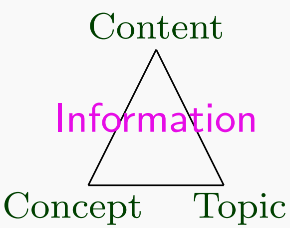
not able to guess meta-data such as quality, timeliness, actuality, ...



Proposal 1: Querying with a Topic-Based, Concept-Backed, User-Oriented CMS

Not trapped in the SQL trap

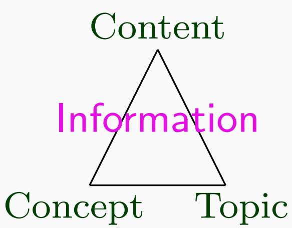




Proposal 2: Graphical Reasoning instead of Logical Reasoning

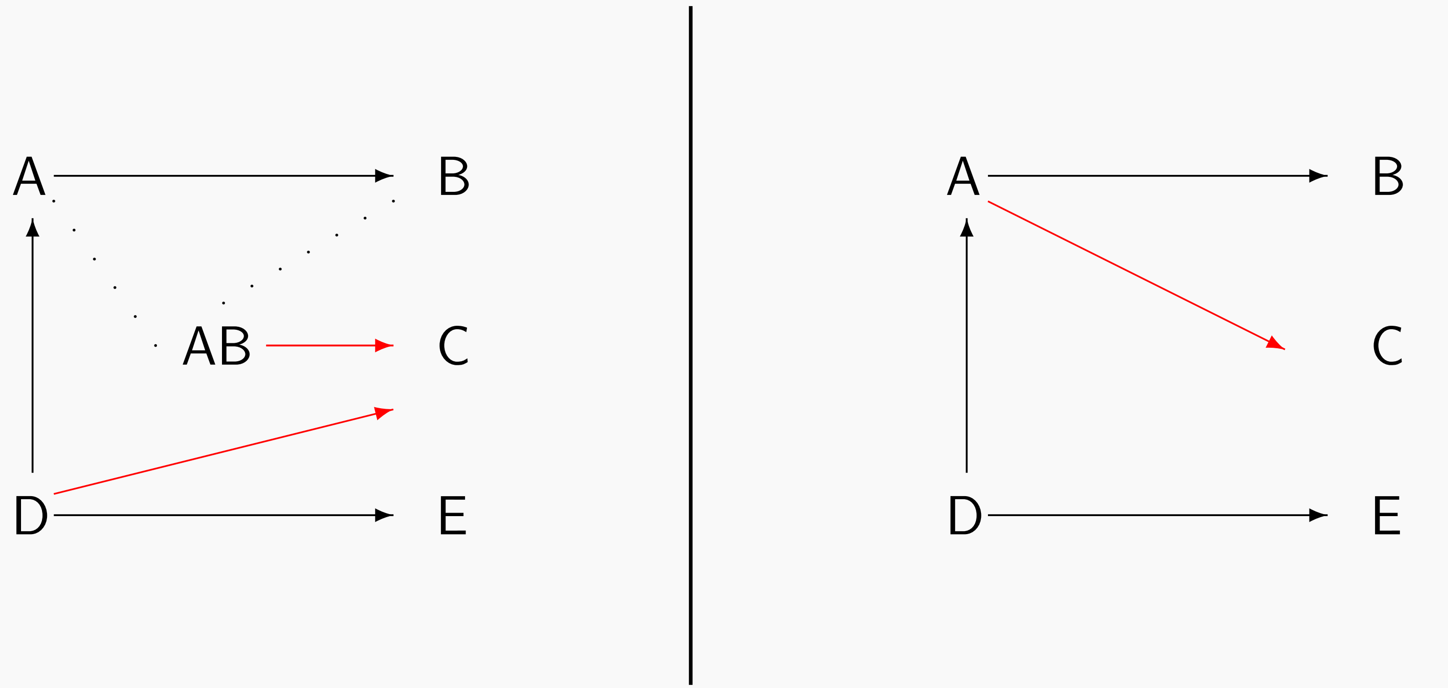
The Power of Visual Reasoning

- Human reasoning is also spacial;
  many useful concepts, e.g., "behind", "far", "easy to reach", ...
- Graphical presentation uses our second language;
  years before we learn to write
  far more expressive than natural language;
- Visual reasoning also uses allegories, signs, ..., metaphors
- Sequential representation is a difficult matter;
- Mathematics & logics teach however linear reasoning
  i.e., we need to learn and to adapt ourselves



The Simplicity of Graphical Reasoning

Given two FD sets.

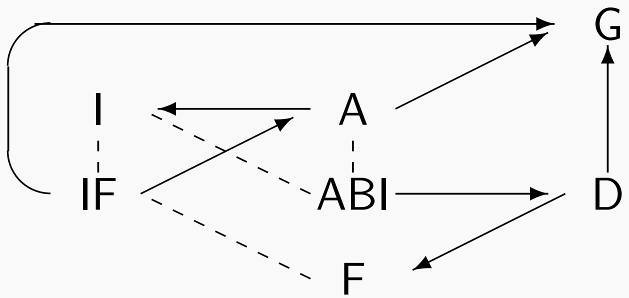


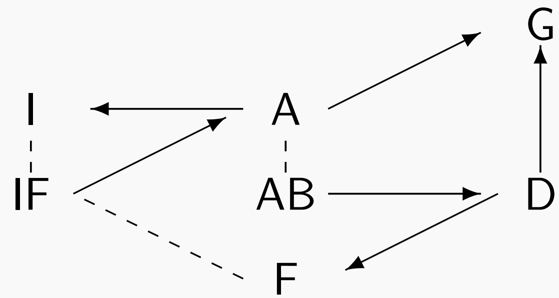
`Are the FD sets equivalent??`

The Simplicity of Graphical Reasoning

$U_R = \{A, B, D, F, G, I\}$

$\sigma_R = \{A \longrightarrow IG, D \longrightarrow FG, IAB \longrightarrow D, IF \longrightarrow AG\}$





Classical synthesis algorithms:

$$R_1 = (\{A, G, I\}, \{A \longrightarrow GI\})$$

$$R_2 = (\{A, F, I\}, \{A \longrightarrow I, FI \longrightarrow A\})$$

$$R_3 = (\{A, B, D\}, \{AB \longrightarrow D\})$$

$$R_4 = (\{D, F, G\}, \{D \longrightarrow FG\})$$

This normalisation not minimal! Although normalisation theory teaches so!

Instead of R_1 take $R_1' = (\{A, G\}, \{A \longrightarrow G\})$.

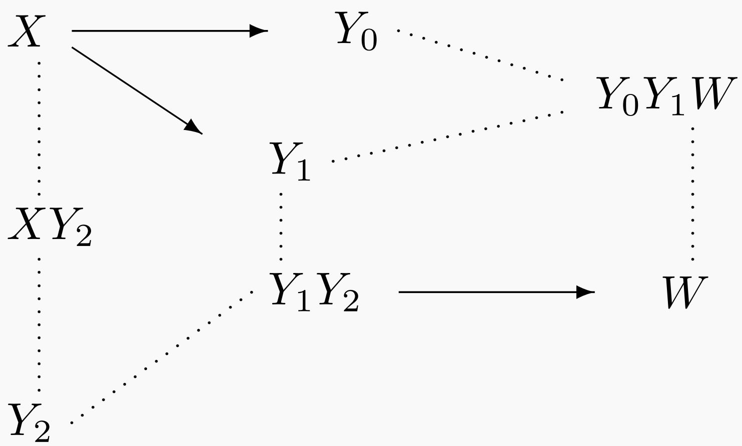
R_2 is not in BCNF. It cannot be split into two relation schemata.

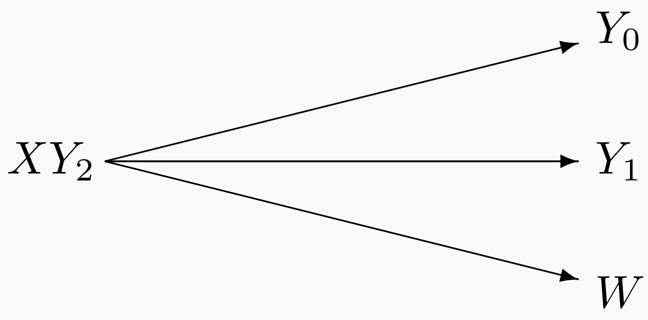
The Simplicity of Graphical Reasoning

Darwen FD rule

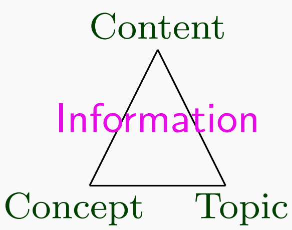
$$\frac{X \to Y_0Y_1,\ Y_1Y_2 \to W}{XY_2 \to Y_0Y_1W}$$

Is the rule correct?





State-of-art
Systematics
VisualSQL
Query forms
Answer forms
Question2Query
Liquefaction
Conclusion



Axiomatisation for Functional Dependencies for Visual Reasoning

with singleton right sides

$$(S) \quad \frac{Y \rightarrow B}{YC \rightarrow B}$$

$$(T) \quad \frac{Y \rightarrow A,\ YA \rightarrow B}{Y \rightarrow B}$$

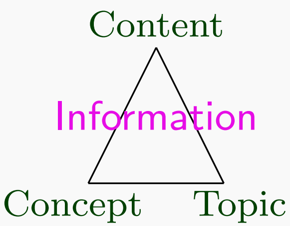
$$(P) \quad \frac{YC \not\rightarrow B}{Y \not\rightarrow B}$$

$$(Q) \quad \frac{Y \rightarrow A,\ Y \not\rightarrow B}{YA \not\rightarrow B}$$

$$(R) \quad \frac{YA \rightarrow B,\ Y \not\rightarrow B}{Y \not\rightarrow A}$$

$$(\square) \quad \neg(Y \rightarrow B,\ Y \not\rightarrow B)$$

Also for negated functional dependencies.



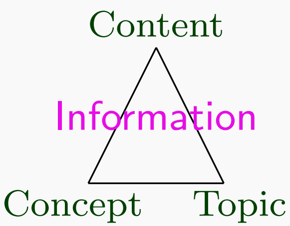
Proposal 3: Graphical Querying together with NoSQL for Big Data

Large data, very large data, huge data, big data: all the time the same problem (limited resources), i.e. volume; nowadays also velocity, variety, and veracity (dependability, limited quality and viability) [the four big data V's]

From NO!!!SQL to Not-only-SQL: for advanced data sets, integrated query languages

Schema-less computation: currently without schema, next with associating schemata

XML has solved all our problems: statement since 18 years, not yet true; a lot of research for ill-defined languages such as XPath



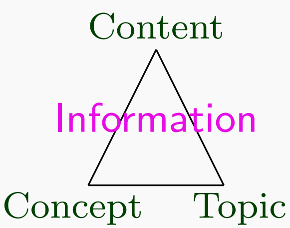
Systematical Querying

Traditional database querying

$$\begin{array}{rcl}
\texttt{input} & : & (\text{DBMS query form}\ ,\ \text{database schema}) \mapsto \text{SQL query} \\
\texttt{process} & : & \text{SQL query} \mapsto \text{SQL answer set} \\
\texttt{output} & : & \text{SQL answer set} \mapsto \text{DBMS answer representation}
\end{array}$$

Linguistic search facilities

$$\begin{array}{rcl}
\texttt{map} & : & \text{search concept} \mapsto \text{query form} \\
\texttt{compile} & : & (\text{query form}\ ,\ \text{database schema}) \mapsto \text{SQL query} \\
\texttt{map} & : & \text{result concept} \mapsto \text{answer form} \\
\texttt{process} & : & \text{SQL query} \mapsto \text{SQL answer set} \\
\texttt{output} & : & (\text{SQL answer set}\ ,\ \text{answer form}) \mapsto \text{answer to search}
\end{array}$$



Kinds of Search Features Applicable to Types

Search by main properties: weighted high in the star schema
the classical SQL capability

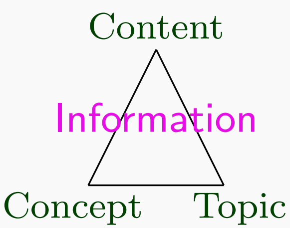
Fuzzy search generalization of domain values and similarity values. *SoundEx*

Search by associations: step-wise scoping, refinement and narrowing; its context

Search by meta-properties: space, time, history of objects and database, profiles of actors, specific data types, specific constraints

Search on the basis of the utilization record: search engine records results of previous search request, the story space of a group of users or log file

Search through browsing: the entire set of objects is scanned on the basis of some main properties



Search Combined with Control Approaches

- *depth-first search* (developing each type completely before moving to the next type),
- *hill-climbing search* (using a selection function and a heuristic function in order to determine the next best local step),
- *breadth-first search* (developing all types to a certain extent before moving to the next reification),
- *beam-search strategy* (same procedure as breadth-first-search but with the use of heuristic functions to select the next types), or
- *best-first search* (developing the best unexpanded type as far as possible using a general control function and a general selection function).

Deriving The Navigation Search From Associations

Traverse uni-directional associations

Traverse qualified association

Traverse generalizations/specializations

- Upwards traversal
- Downwards traversal
- Obtain the XML object

Traverse from link to object

Traverse from object to link

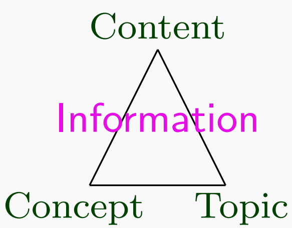
- Link collection
- Traversal by roles

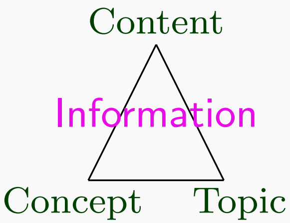
Filter objects

Filter links

Traverse from object to value

Traverse from link to value





Context-based Retrieval from the Web

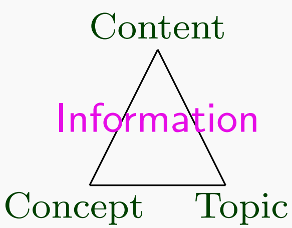
Context capturing performed at the client side software. It is based on correlation-basic metrics for similarity and may use advanced dictionaries, e.g., WordNet.

Keyword extraction from the captured text and context based on clustering algorithms.

High-level classification of the query to a small set of predefined domains. The ontology object may be applied to a set of search engines, may be ranked by their relevance and coverage depending on the keyword set.

Ordering and adhesion of query results is obtained from different search engines by reranking with distance measures, adhesion, and cohesion functions.

Context-based retrieval is a variant of 'blind', non-informed search. It may be enhanced by search algorithms, e.g. the A^*-algorithm.



Visual SQL in a Nutshell

Object-relational diagram with essential types and attributes

Comparison and aggregation operators beside the classical functions of the relational algebra

Views based on a sub-graph representation

Retrieval language using output ticks and sub-diagrams

Update language based on the visual representation

Path language similar to XPath (but on semantically correct grounds)

Fully fledged semantics based on HERM logical calculus

Graphical representation of constraints and their enforcement policy

Potentially explicit representation of trigger suites and stored procedures

IDEF database schema with DBMain

Specific Assumptions of Visual Reasoning

ER schemata are nice but the later transformation to logical schemata contains many assumptions that must be included into the translation procedure of Visual SQL;

IDEF schemata are more convenient since the transformation is already used;

Logical tricks e.g., redundant attributes, must however incorporated;

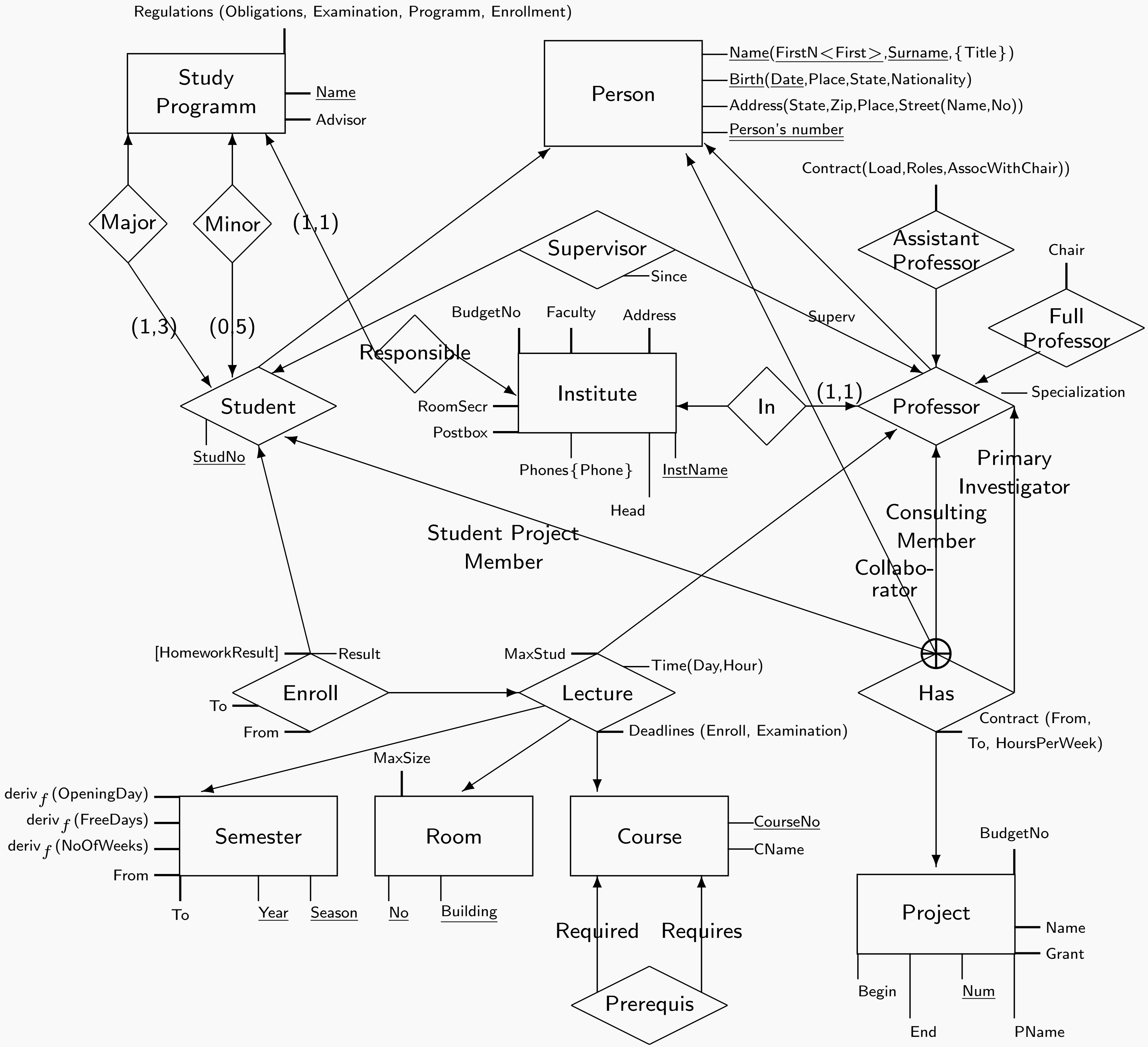
Enhanced IDEF schema seems to be the right compromise;

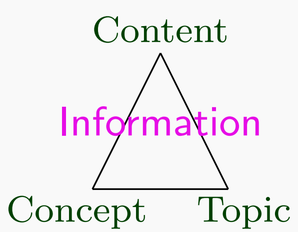
Click and drop is a must;

Select and connect is a must;

Zooming is an essential feature but not yet used, can however be mimicked.

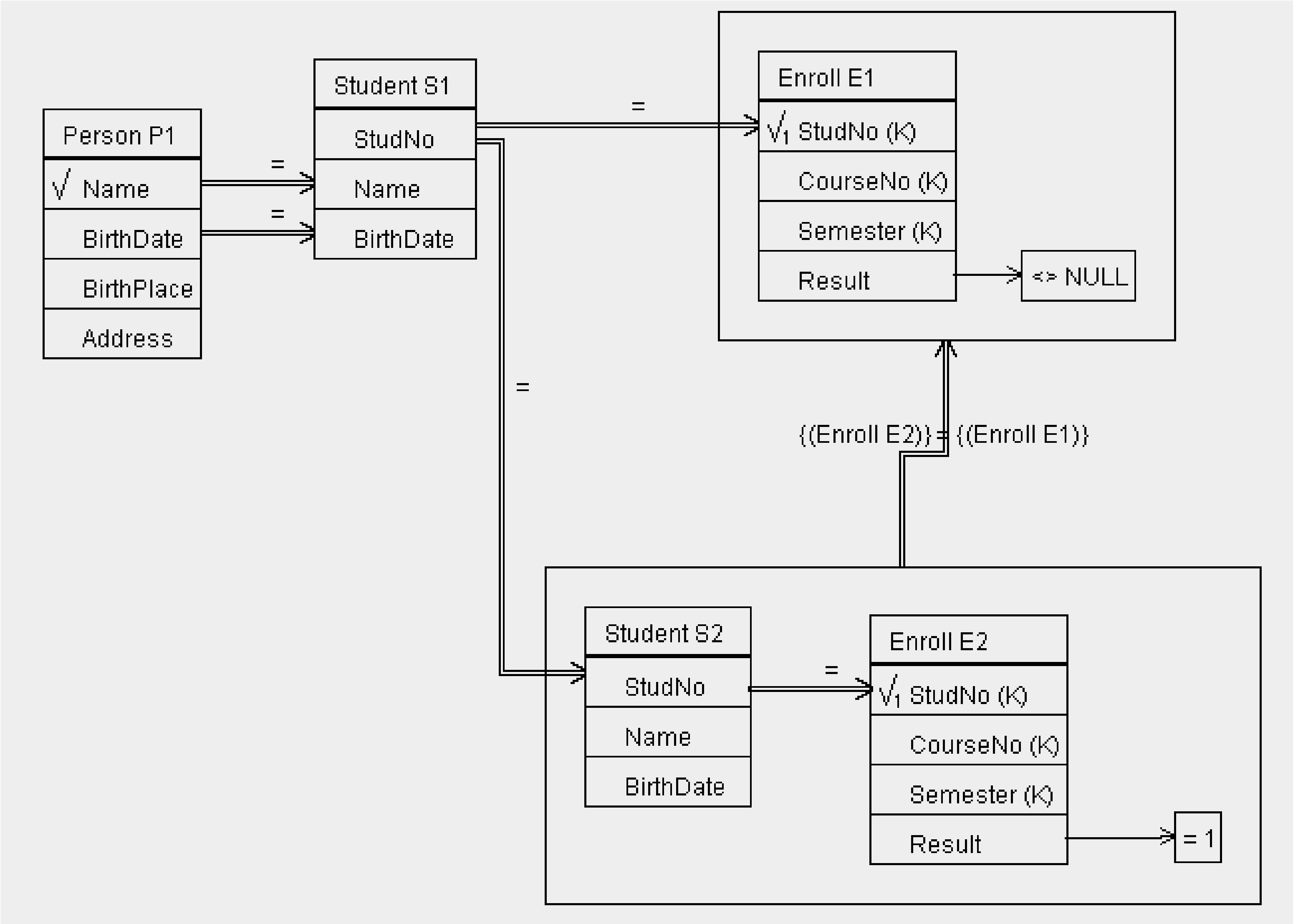
Querying
Answering
Visuality
29.11.2013
B. Thalheim



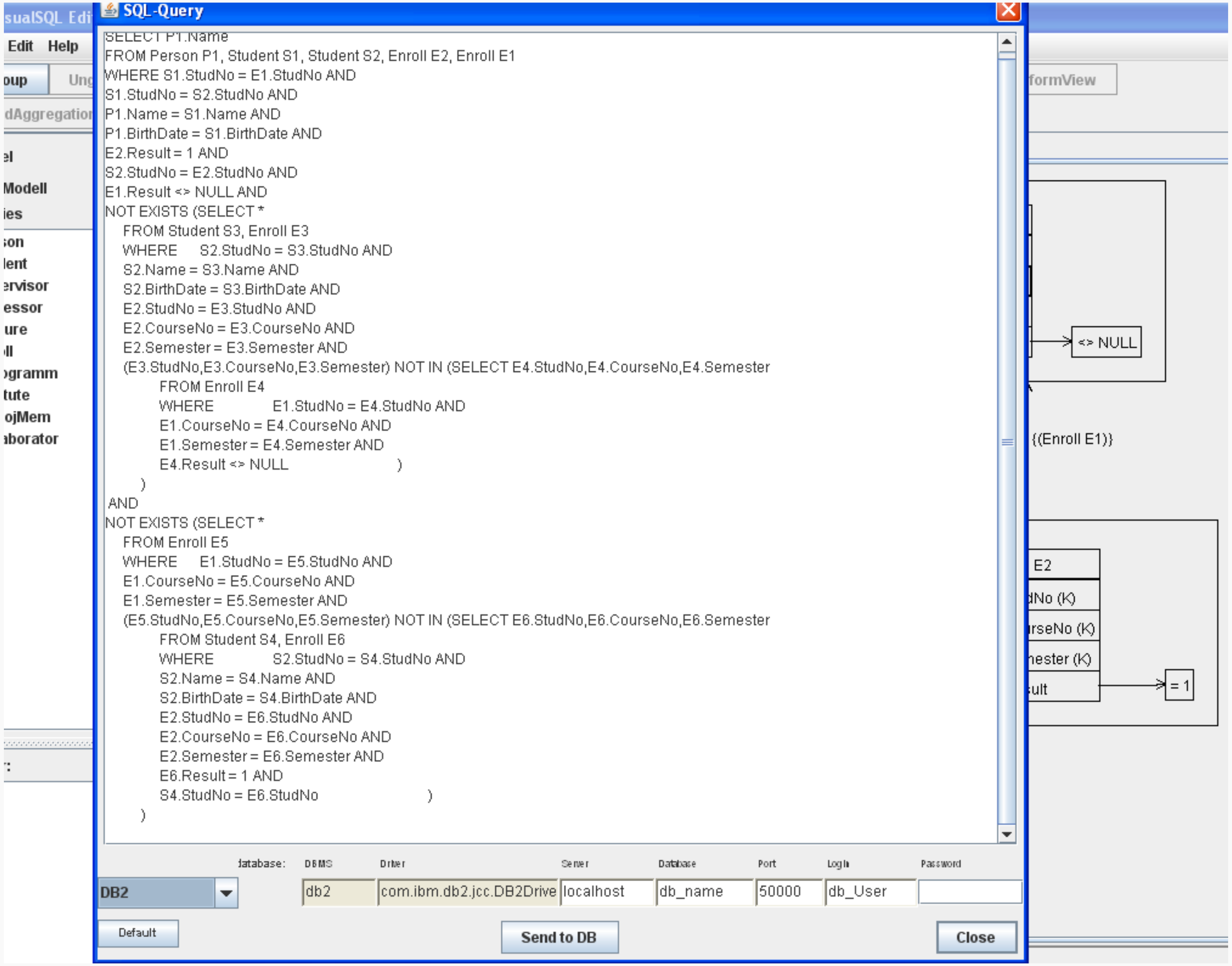


Visual SQL: Our Super-Students

Students that study with excellence, without “misses”



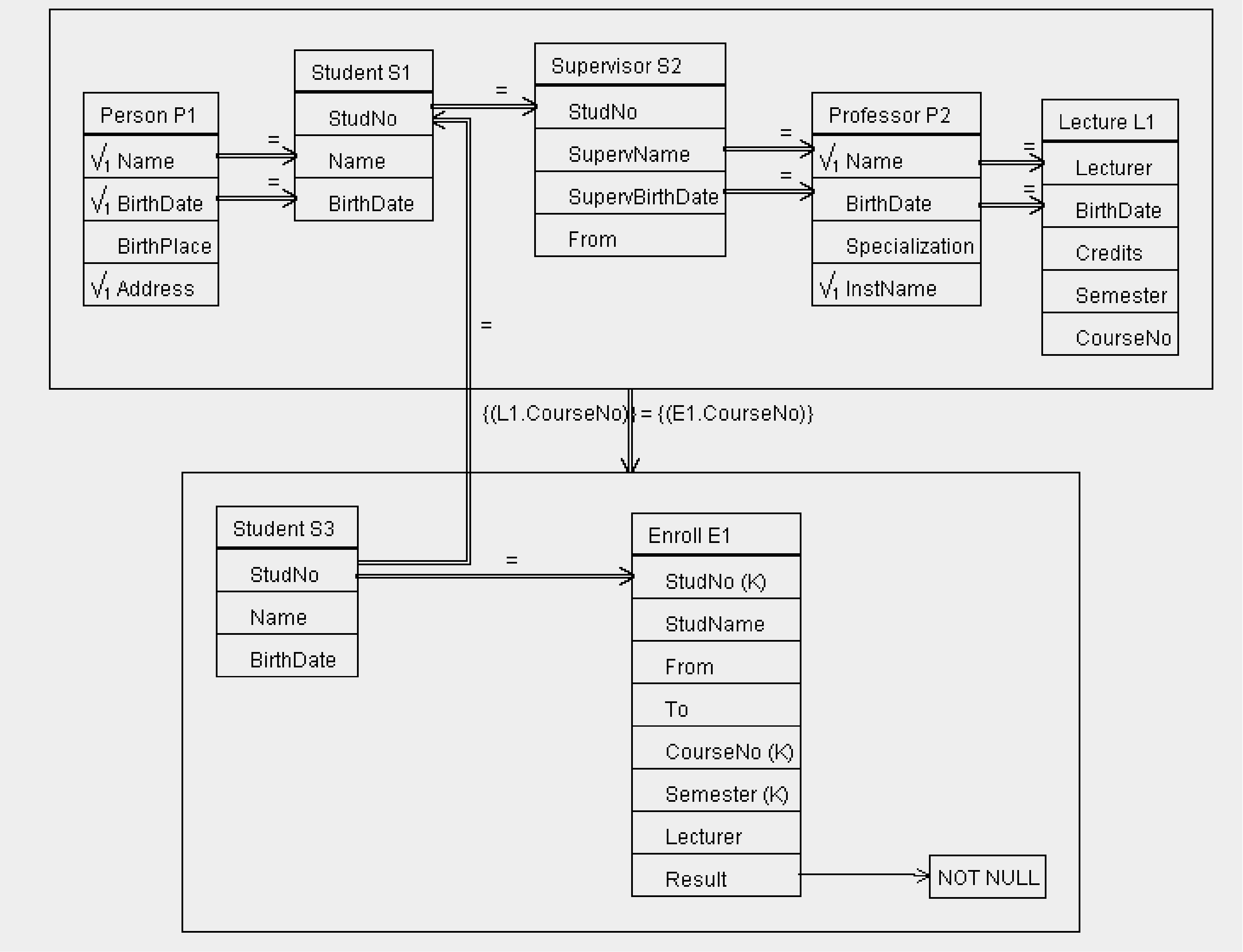
Visual SQL: Our Super-Students



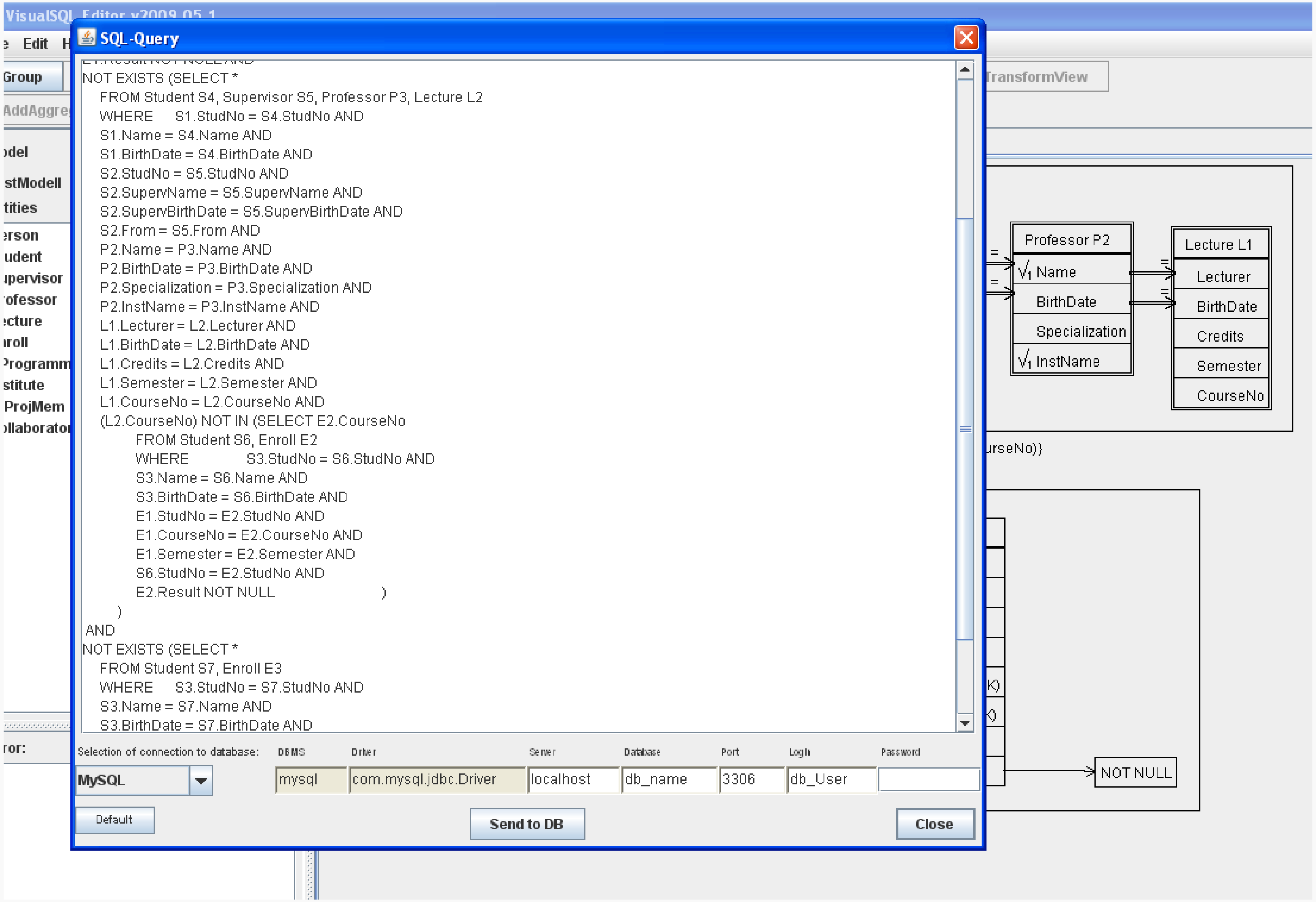
SQL: Students without any “Misses”

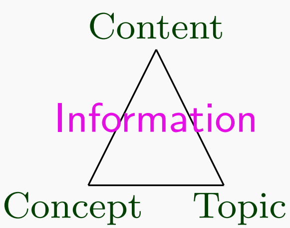
```
SELECT P1.Name
FROM Person P1, Student S1, Student S2, Enroll E2, Enroll E1
WHERE S1.StudNo = E1.StudNo AND S1.StudNo = S2.StudNo AND
  P1.Name = S1.Name AND P1.BirthDate = S1.BirthDate AND
  E2.Result = 1 AND S2.StudNo = E2.StudNo AND E1.Result <> NULL AND
  NOT EXISTS (SELECT *
     FROM Student S3, Enroll E3
     WHERE S2.StudNo = S3.StudNo AND S2.Name = S3.Name AND
      S2.BirthDate = S3.BirthDate AND E2.StudNo = E3.StudNo AND
      E2.CourseNo = E3.CourseNo AND E2.Semester = E3.Semester AND
      (E3.StudNo,E3.CourseNo,E3.Semester) NOT IN (
        SELECT E4.StudNo,E4.CourseNo,E4.Semester
        FROM Enroll E4
        WHERE E1.StudNo = E4.StudNo AND E1.CourseNo = E4.CourseNo AND
           E1.Semester = E4.Semester AND E4.Result <> NULL )
               )
     AND
     NOT EXISTS (SELECT *
      FROM Enroll E5
      WHERE E1.StudNo = E5.StudNo AND E1.CourseNo = E5.CourseNo AND
        E1.Semester = E5.Semester AND
        (E5.StudNo,E5.CourseNo,E5.Semester) NOT IN (
           SELECT E6.StudNo,E6.CourseNo,E6.Semester
           FROM Student S4, Enroll E6
           WHERE S2.StudNo = S4.StudNo AND S2.Name = S4.Name AND
             S2.BirthDate = S4.BirthDate AND E2.StudNo = E6.StudNo AND
             E2.CourseNo = E6.CourseNo AND E2.Semester = E6.Semester AND
             E6.Result = 1 AND S4.StudNo = E6.StudNo )
        );
```

Visual SQL: Total Fans of their Supervisor



Visual SQL: Total Fans of their Supervisor





SQL: Total Fans of their Supervisor

```
SELECT P1.Name,P1.BirthDate,P1.BirthPlace,P1.Address
FROM Person P1, Student S1, Supervisor S2, Professor P2, Lecture L1, Student S3, Enroll E1
WHERE S3.StudNo = S1.StudNo AND P1.Name = S1.Name AND
  P1.BirthDate = S1.BirthDate AND S1.StudNo = S2.StudNo AND
  S2.SupervName = P2.Name AND S2.SupervBirthDate = P2.BirthDate AND
  P2.Name = L1.Lecturer AND P2.BirthDate = L1.BirthDate AND
  S3.StudNo = E1.StudNo AND E1.Result NOT NULL AND
  NOT EXISTS (SELECT *
    FROM Student S4, Supervisor S5, Professor P3, Lecture L2
    WHERE S1.StudNo = S4.StudNo AND S1.Name = S4.Name AND
      S1.BirthDate = S4.BirthDate AND S2.StudNo = S5.StudNo AND
      S2.SupervName = S5.SupervName AND S2.SupervBirthDate = S5.SupervBirthDate AND
      S2.From = S5.From AND P2.Name = P3.Name AND P2.BirthDate = P3.BirthDate AND
      P2.Specialization = P3.Specialization AND P2.InstName = P3.InstName AND
      L1.Lecturer = L2.Lecturer AND L1.BirthDate = L2.BirthDate AND
      L1.Credits = L2.Credits AND L1.Semester = L2.Semester AND
      L1.CourseNo = L2.CourseNo AND (L2.CourseNo) NOT IN (
        SELECT E2.CourseNo
        FROM Student S6, Enroll E2
        WHERE S3.StudNo = S6.StudNo AND S3.Name = S6.Name AND
          S3.BirthDate = S6.BirthDate AND E1.StudNo = E2.StudNo AND
          E1.CourseNo = E2.CourseNo AND E1.Semester = E2.Semester AND
          S6.StudNo = E2.StudNo AND E2.Result NOT NULL )
    )
  AND
  NOT EXISTS (SELECT *
    FROM Student S7, Enroll E3
    WHERE S3.StudNo = S7.StudNo AND S3.Name = S7.Name AND
      S3.BirthDate = S7.BirthDate AND E1.StudNo = E3.StudNo AND
      E1.CourseNo = E3.CourseNo AND E1.Semester = E3.Semester
      );
```

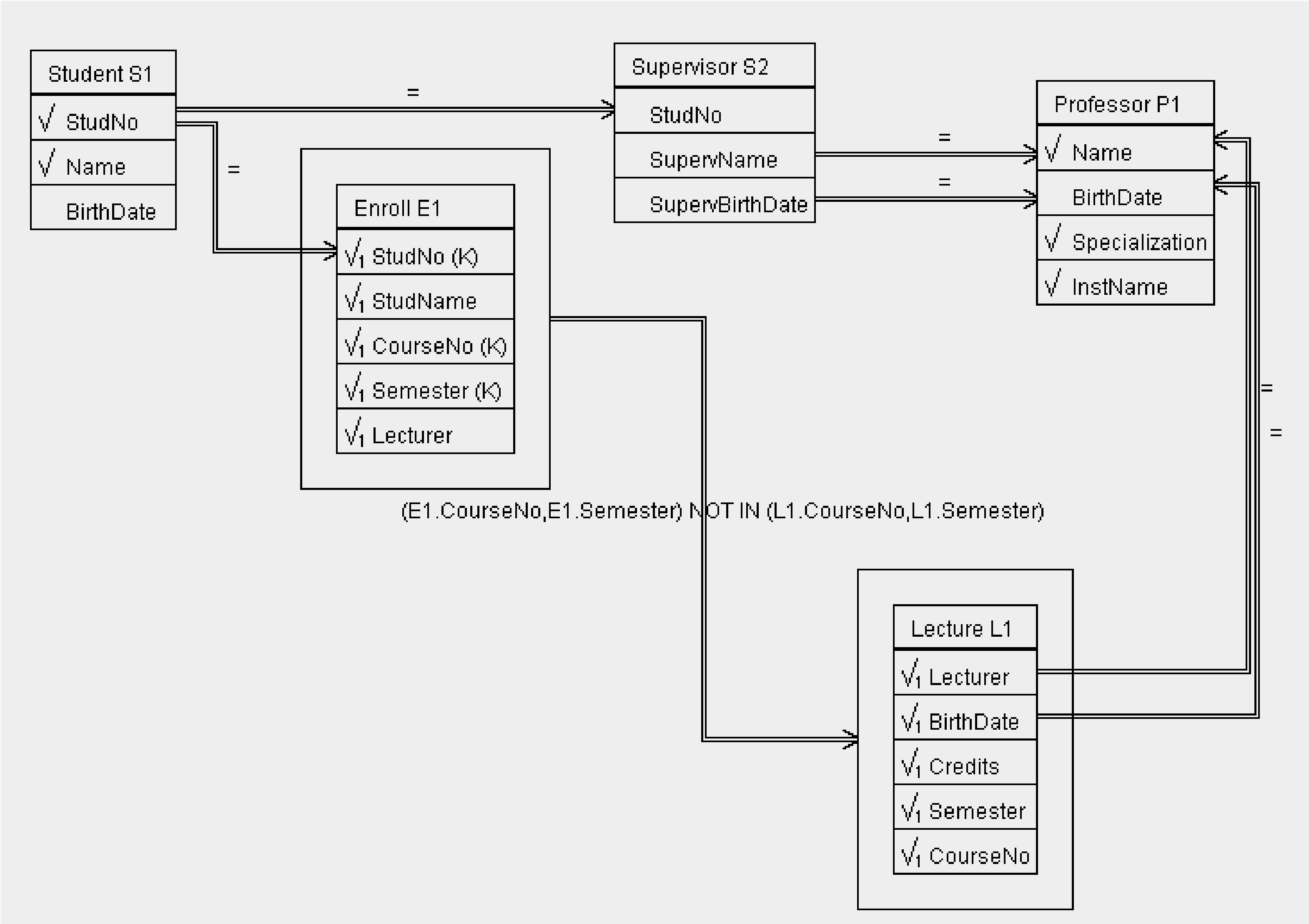
Do we have Time for an Interrupt?

It might be now time to play a bit!

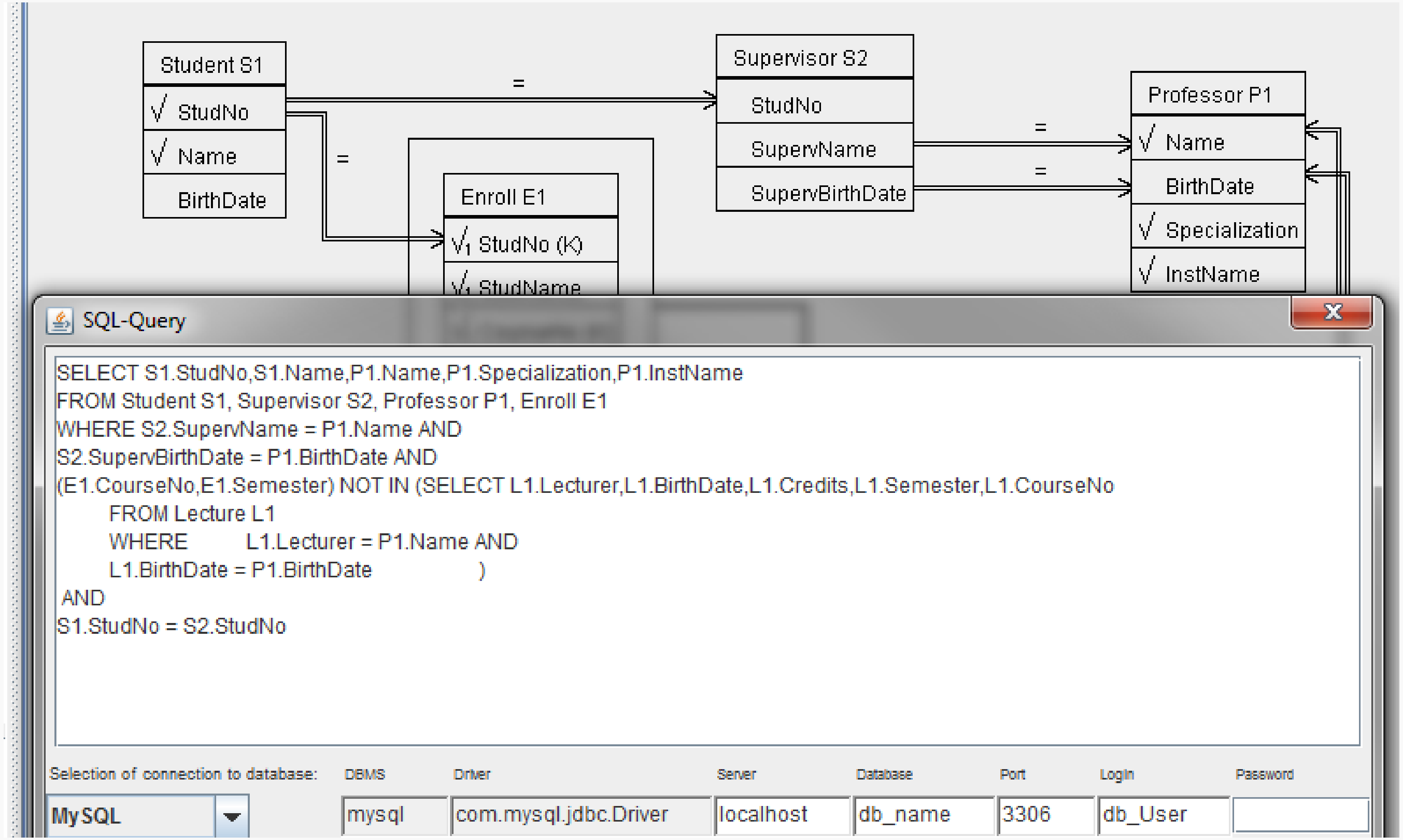
Which students are anti-fans of their supervisors?

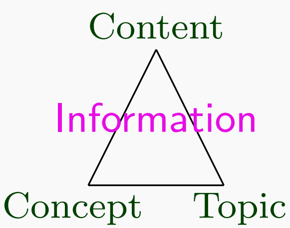
Try it with XPath!

If there is no time for playing: Anti Fans of their Supervisors



If there is no time for playing: Anti Fans of their Supervisors





Visual SQL Translation Profiles

HERM: one extended ER model that supports compact representation and has a well-defined semantics

Object-relational model: ID-based treatment with complex attributes (reference values, structured values, collections (finite sets, finite lists, arrays)), reference semantics, behavior based on methods

Relational model: atomic attributes, relations, complex constraint treatment

SQL-92 model: atomic attributes, tables, restricted constraint treatment

Aim for Visual SQL mapping

to SQL-92 (e/i/f), SQL:1999, SQL:2003 mapping

- homogeneous
- bijective mapping
- for all types

Mappings Consider

Treatment of hierarchies

Controlled redundancy with corresponding functionality

Null and default values support restricting functionality of types

Enforcement of constraints beside key and domain constraints

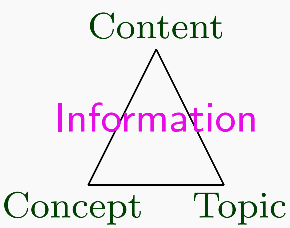
Naming conventions and abbreviation rules

Set or pointer semantics

Utilisation of weak types

Translation of complex attributes

Global or type-wise translation



Treatment of Hierarchies

Event non-separation approach: Types are separated from their subtypes.

  class inclusion constraints

Event separation approach: Hierarchy is partitioned into disjoint types.

  object belongs either to one or more of the subtypes or it belongs to the supertype and none of its subtypes

  exclusion constraints

Union approach: The hierarchy is merged into one type.

  additional attributes for type information

Universal relation approach: union approach + embedding relationship types

Generalisation and specialisation

Strong specialisation: Subtypes have their specific attributes and inherit one key from the supertype

Strong generalisation: Subtypes have all attributes.

  supertype has only the common key attributes and attributes specific for the supertype

Mixed approach

Default Translation Options Used

- *Event non-separation approach*
- *Strong specialisation* for unary relationship types and *strong generalisation* for cluster types
- *No redundancy* in types except referential constraints
- *Null value support* for all attributes which are not bounded through attribute inheritance
- *Enforcement of constraints* on the basis of *declarative approaches* if possible
- *Component inclusion constraints* on a declarative basis
- Application of *naming conventions*
- *Identification extension* whenever key attributes become too complex
- *Invariance* of complex attributes

CASE tools have their own default profile.

Translation Profile for Visual SQL to SQL-92

- Role extension whenever names clash
- Variables are only used if they are introduced in Visual SQL
- Additional attributes
- Shortening of labels
- Blocks as subqueries
- Set containment through `(NOT) EXIST` or `(NOT) IN`
- Integrity constraints are either mapped to declarative constraints or triggers (depending on the DBMS)
- ID extension if required by the DBMS, e.g., Oracle

Advantages of Visual SQL

Visual SQL as a database description language

Visual SQL is more natural and fits better to linguistic environments

Syntactic and semantic quality raises for **complex** queries

Object-relational technology can be better treated on the basis of Visual SQL

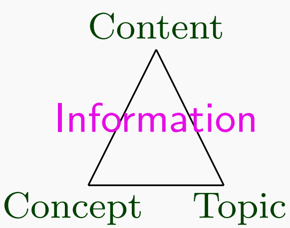
Simple maintenance and correction of query formulations

Easy correction and trace of errors in queries

SQL to Visual SQL translation

Database tuning with Visual SQL

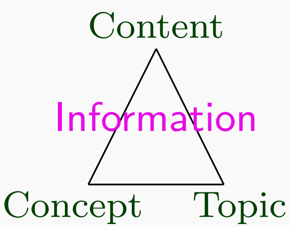
Global constraint maintenance



Problems with SQL Representation?

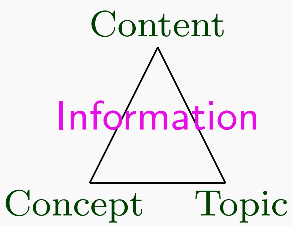
- Why it is sometimes so difficult to transform our question to SQL?
- Why the user has to learn the (logical) database schema?
- Why we should not use the natural language form for query formulation?
- If there are reasons for non-use: Is there a fragment of natural language we might use?
- Are we able to support at least Indoeuropean languages?
- Why we should not use the users expectation for answer formatting?
- What is the content of a question?
- What is the expected answer?

State-of-art
Systematics
VisualSQL
Query forms
Answer forms
Question2Query
Liquefaction
Conclusion



The W^7(+W^4+W^{17}H) Question Frame

- *matter* (what, concepts, in what way)
- *situational context* (when, where, in what means).
- *user profile* (who) and *user portfolio* (wherefore, wherein, where, for what, wherefrom, whence, what)
- *carrier language* (wherewith) within a *namespace* (whereto, by what means).
- *answer solution characteristics* (how, why, whereto, when, for which reason),
- *solution context embedding* (whereat, whereabout, whither, when)
- *surplus value* (worthiness) of the answer.



Query Forms

The more general and far simpler form of queries

(question content, matter (concepts, situation), user(profile, portfolio), carrier).

parametric view expression $expr(T_1, ..., T_n, x_1, ...x_m)$

graph of query notions

graph can be extended to the given DB schema through homomorphic embedding

definable by Visual SQL

- see example below

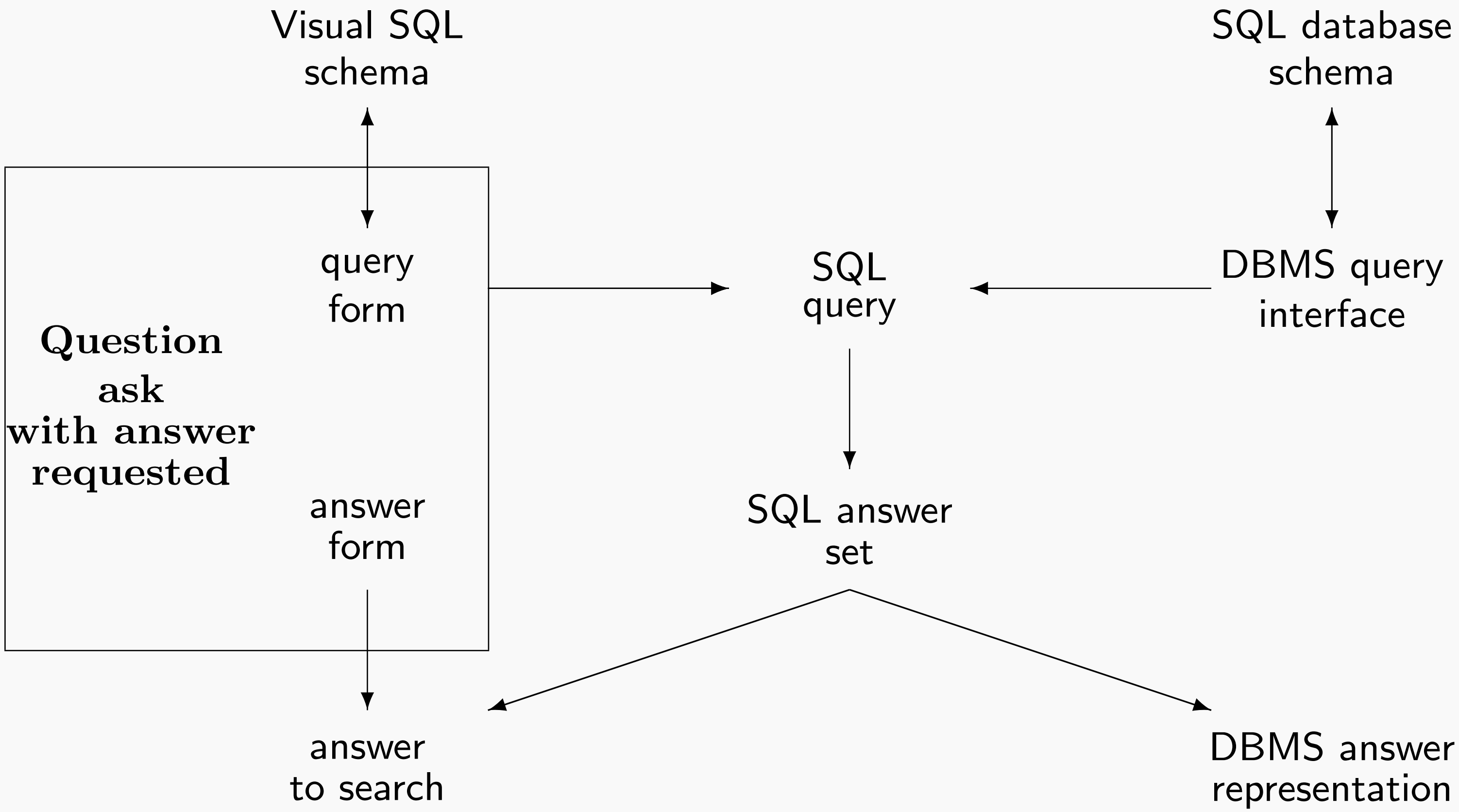
embedding through graph grammar formalism

- with integrity constraints

Answer Forms

Any question contains also the expected answer format.

(answer content, solution (characteristics, context, value))



Six Steps From Question to Query

(1) Extension of the Search Question

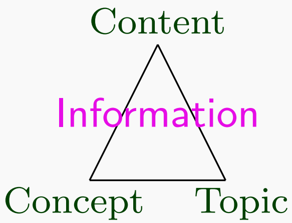
(2) Orthonormalisation and Extension of the Search Question and Mapping to Query Forms and Answer Forms

(3) Rephrasing of the Question into an Existential Form

(4) Mapping of the Query Form to Database Schema Notions

(5) Derivation of the Extended Answer Form

(6) Derivation of the Database Query

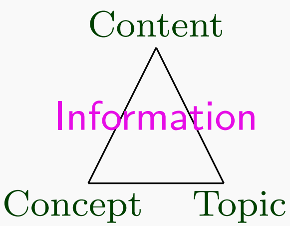


1: Extension of the Search Question

Which students occur only together?

- extend by issuer's context,
- extend by community of practice common sense,
- resolve ambiguities,
- use issuer semantics, e.g. for connectives,
- resolve ellipses, and
- add scope and issuers.

... doing same things at the same time and with the same success?

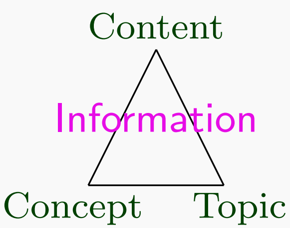


2: Orthonormalisation and Extension of the Search Question and Mapping to Query Forms and Answer Forms

Which students occur only together?

- categorise by the W^7(+W^4+W^{17}H) frame
- orthonormailsation
- connectives interpretation
- abbreviations
- matter, own concepts, aggregates
- profile and portfolio of the issuer, data on demand as information demand
- query form graph
- answer forms graph

Which students complete the same courses in the same term?



Connectives and Quantifiers in Reality?

Different truth definitions

Material, logical, and normative connectives , e.g. implication

- $\psi \rightarrow \phi$ means ϕ necessarily if ψ (strict, logical)
- $\psi \Rightarrow \phi$ means 'It is the case that if ψ (can be observed) then also ϕ.' (material)
- $\psi \supset \phi$ means 'In situations for which there exists a dependence then ϕ follows from ψ (norms) (counter-example-based)

ψ	ϕ	$\psi \rightarrow \phi$, $\psi \Rightarrow \phi$	$\psi \supset \phi$
1	1	1	1
1	0	0	0
0	1	1	??
0	0	1	??

Generalisation operators e.g. (t,f)-quantifier $Q_{t,f}$ with validity dependence of $Q_{r,s}\alpha(x)$ in structure $\mathcal{A}$ such that

$$|\{o \in \pi_x(dom(\mathcal{A})) \,|\, I_x^o(\alpha) = 1\}| = t \text{ and}$$
$$|\{o \in \pi_x(dom(\mathcal{A})) \,|\, I_x^o(\alpha) = 0\}| = f$$

classical $\forall \equiv Q_{*,0}$, $\exists \equiv Q_{t,*}$ for $t \geq 1$, $Majority \equiv Q_{n+k,n}$, $k, n \in \mathbb{N}^+, k \geq 1$

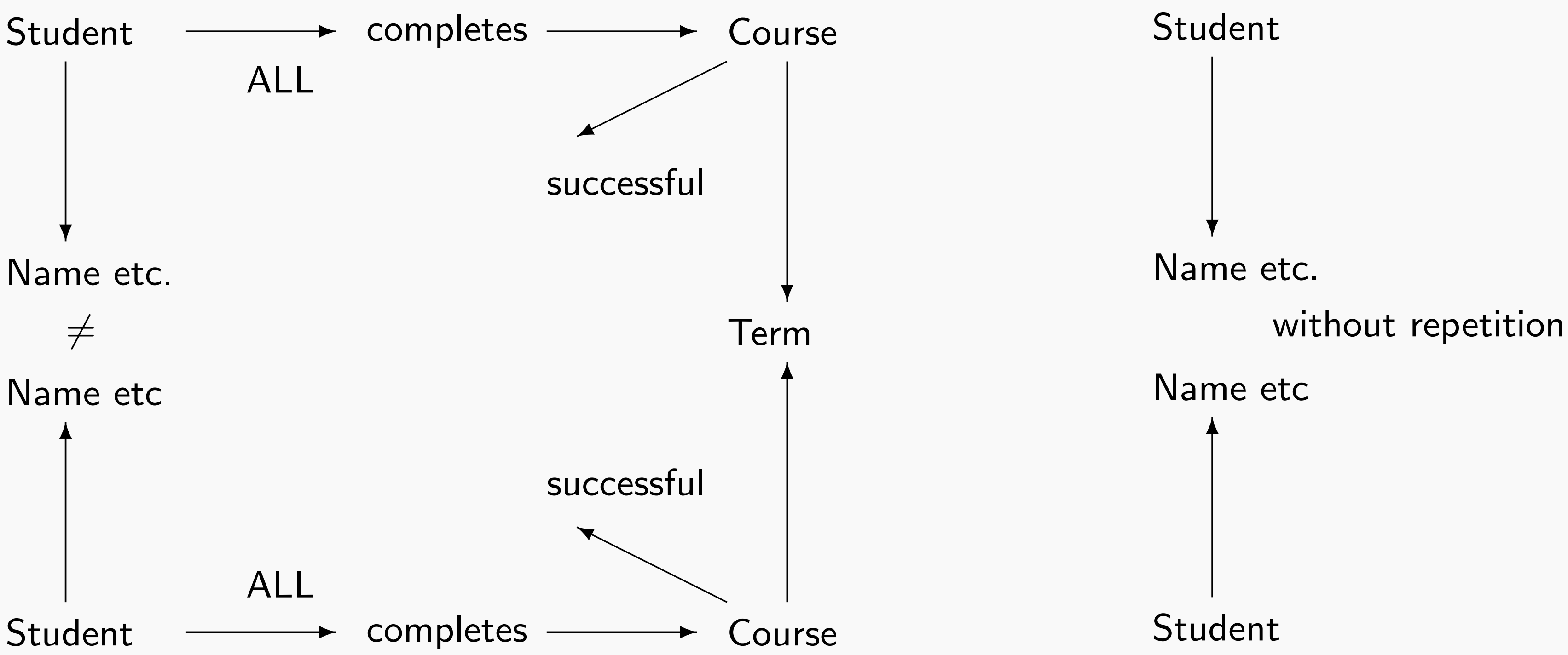
Models for the knowledge operator K_A for actors A

Query and Answer Forms

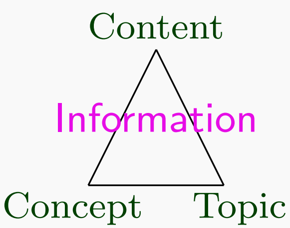
Which students occur only together?

Query form

Answer form



Query and answer forms as potentially homomorphically embedable graph.



3: Rephrasing of the Question into an Existential Form

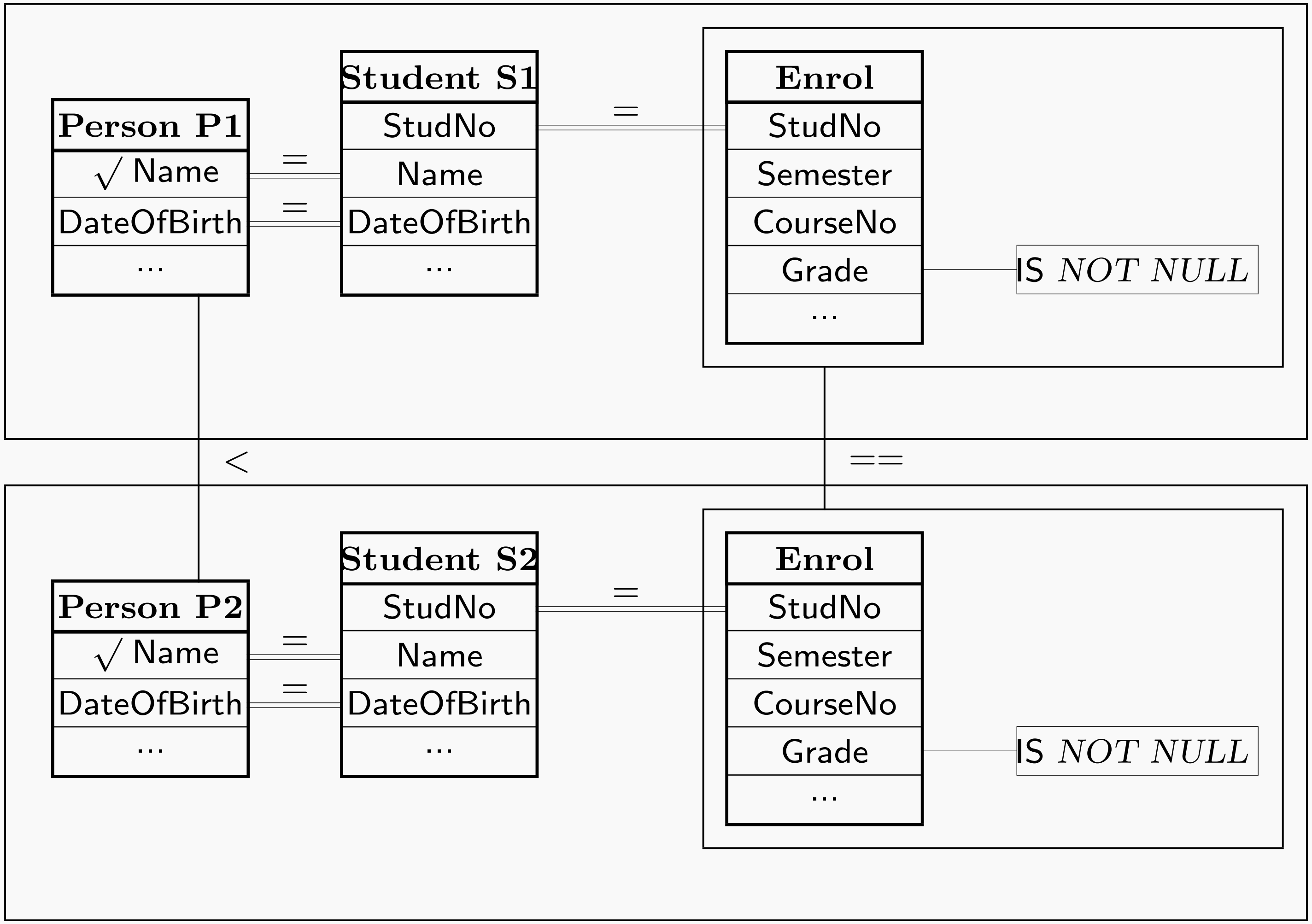
Which students occur only together?

- $\forall$-sentence transformation
  $\neg\exists v((\neg Enrol(a,v) \vee \neg Enrol(b,v)) \wedge (Enrol(a,v) \vee Enrol(b,v)))$
  $=$
  $\forall v((Enrol(a,v) \wedge Enrol(b,v)) \vee (\neg Enrol(a,v) \wedge \neg Enrol(b,v)))$
- connectives transformation
- handling negation
- canonical set representation
- injection into query and answer forms
- null value resolution

... so that does not exist a course that is not taken by the other?

4: Mapping of the Query Form to Database Schema Notions.

Which students occur only together?



far simpler and easier to formulate, to capture, to understand

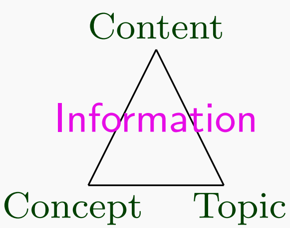
without the SQL burden

5: Derivation of the Extended Answer Form

Which students occur only together?

- parameterisation
- storage alternatives
- answer representation style, e.g., Venetian blind for XML
- add context
- map tu question issuer's language
- extend by features for visualisation, representation
- provide functions for marking, drill-down, roll-up, slice, dice, rotate, refinement, new query issuing, export, session storage, and reuse

... nonsymmetric name pairs ordered by corresponding StudNo...



6: Derivation of the Database Query

Which students occur only together?

- correct formulation
- consider the kind of SQL
- adapt to DBMS profile, facilities
- provide query hints
- derive query using integrity constraints
- consider DBMS and user-defined types
- decompose - if necessary - to views and combination query
- handle NULL
- consider materialisation of sub-results for answer form instantiation

... see next slide ...

The Resulting Query

```
SELECT P1.Name, P2.Name
   FROM Person P1, Person P2, Student S1, Student S2, Enrol H1, Enrol H2
   WHERE P1.Name = S1.Name AND P1.DateOfBirth = S1.DateOfBirth AND
       S1.StudNo = H1.StudNo AND H1.Grade IS NOT NULL AND
       P2.Name = S2.Name AND P2.DateOfBirth = S2.DateOfBirth AND
       S2.StudNo = H2.StudNo AND H2.Grade IS NOT NULL
       AND NOT EXISTS
           (SELECT *
           FROM Enrol H3
           WHERE H3.Grade IS NOT NULL AND
               H3.StudNo NOT IN
               (SELECT H4.StudNo
               FROM Enrol H4
               WHERE H4.StudNo = H2.StudNo
               AND H4.Grade IS NOT NULL)
               AND H1.StudNo = H3.StudNo)
       AND NOT EXISTS
           (SELECT *
           FROM Enrol H5
           WHERE H5.Grade IS NOT NULL AND
               H5.StudNo NOT IN
               (SELECT H6.StudNo
               FROM Enrol H6
               WHERE H6.StudNo = H1.StudNo AND H4.Grade IS NOT NULL)
               AND H2.StudNo = H5.StudNo)
       AND S1.StudNo < S2.StudNo
   GROUP BY P1.Name, P2.Name;
```

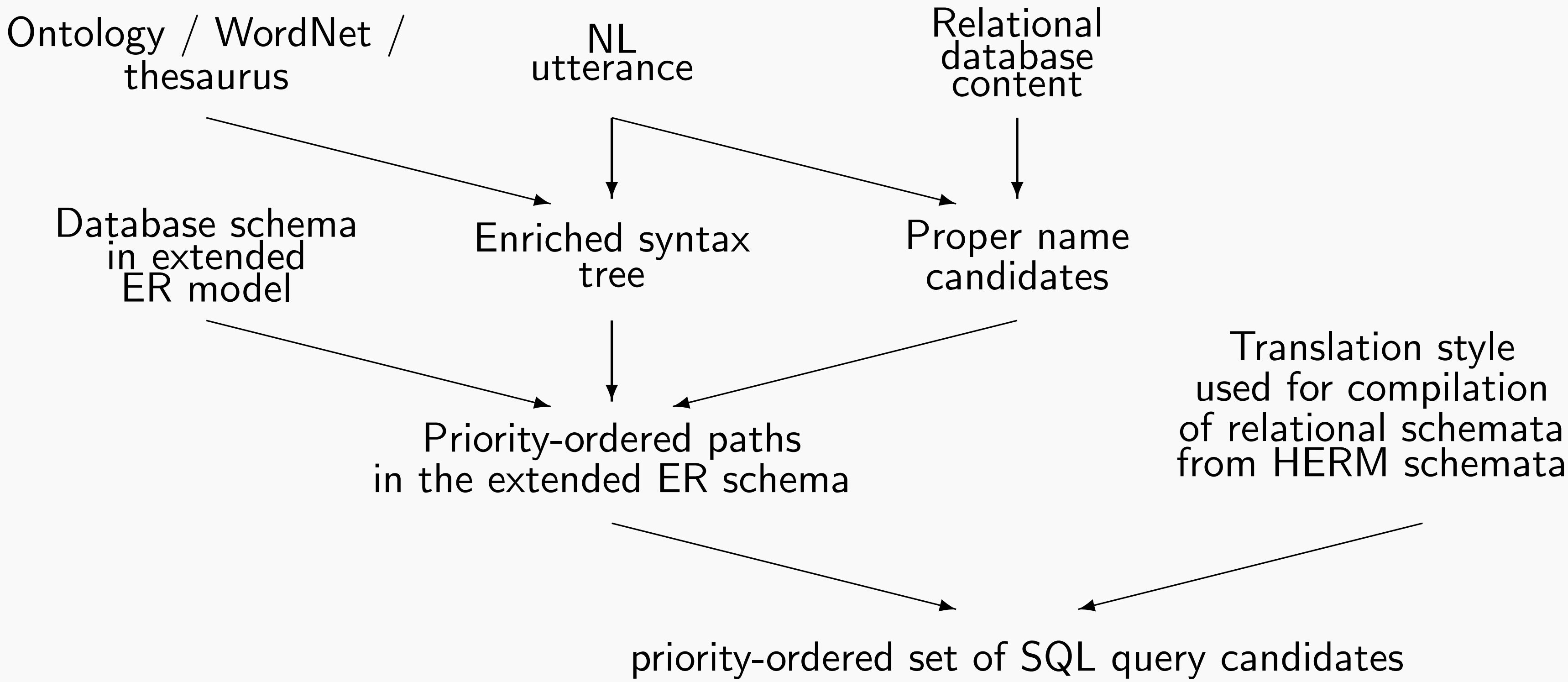
How long would it take you to formulate this query?

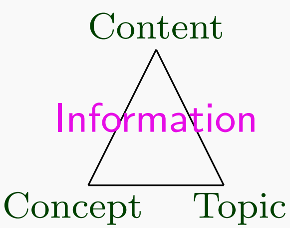
Content
Information
Concept Topic

Question Liquefaction

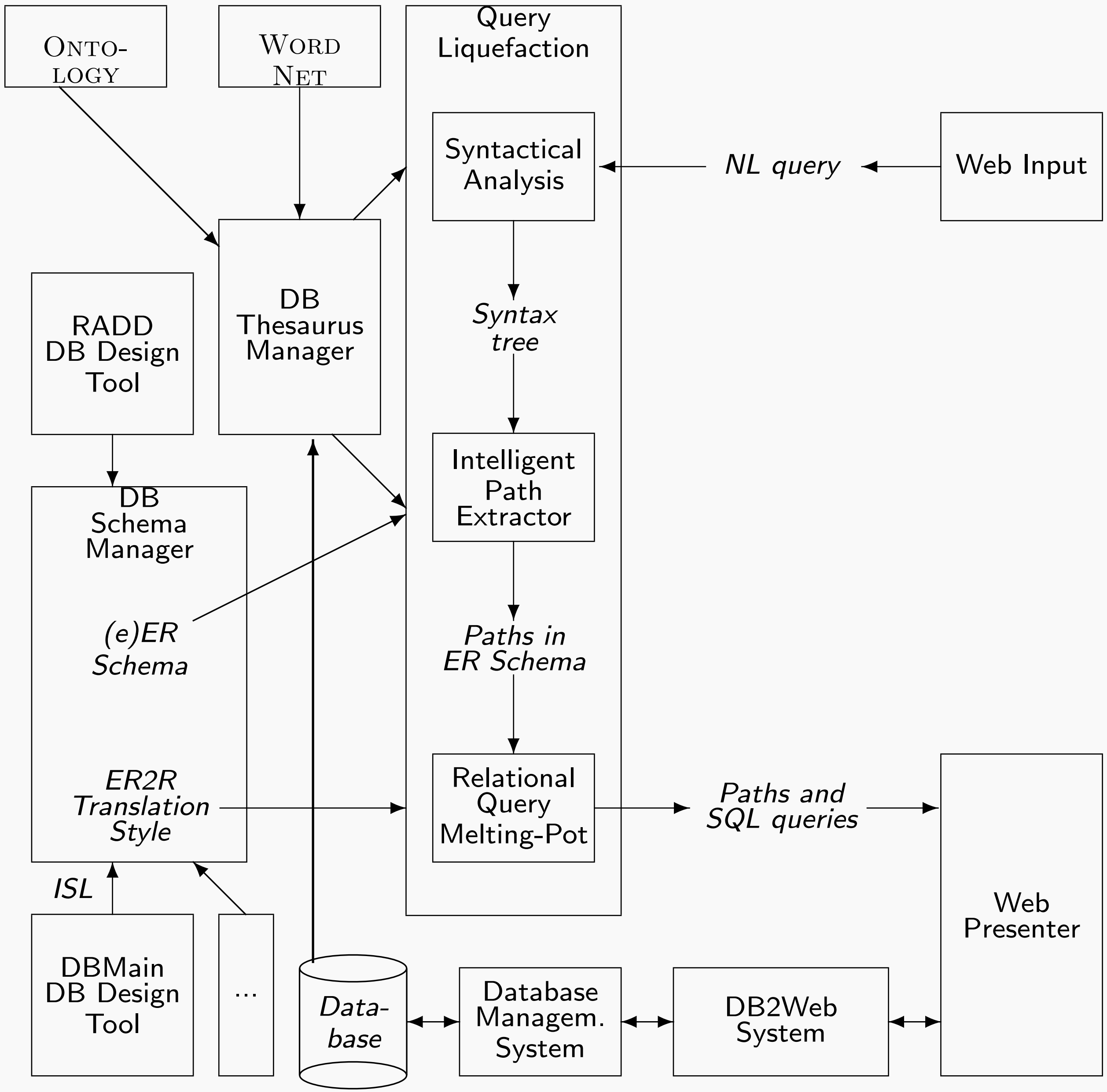
The Three-Step Approach to Automatic SQL Query Generation

Generation of SQL query candidates based on full information





The Cottbus Intelligent NL Request Transformer



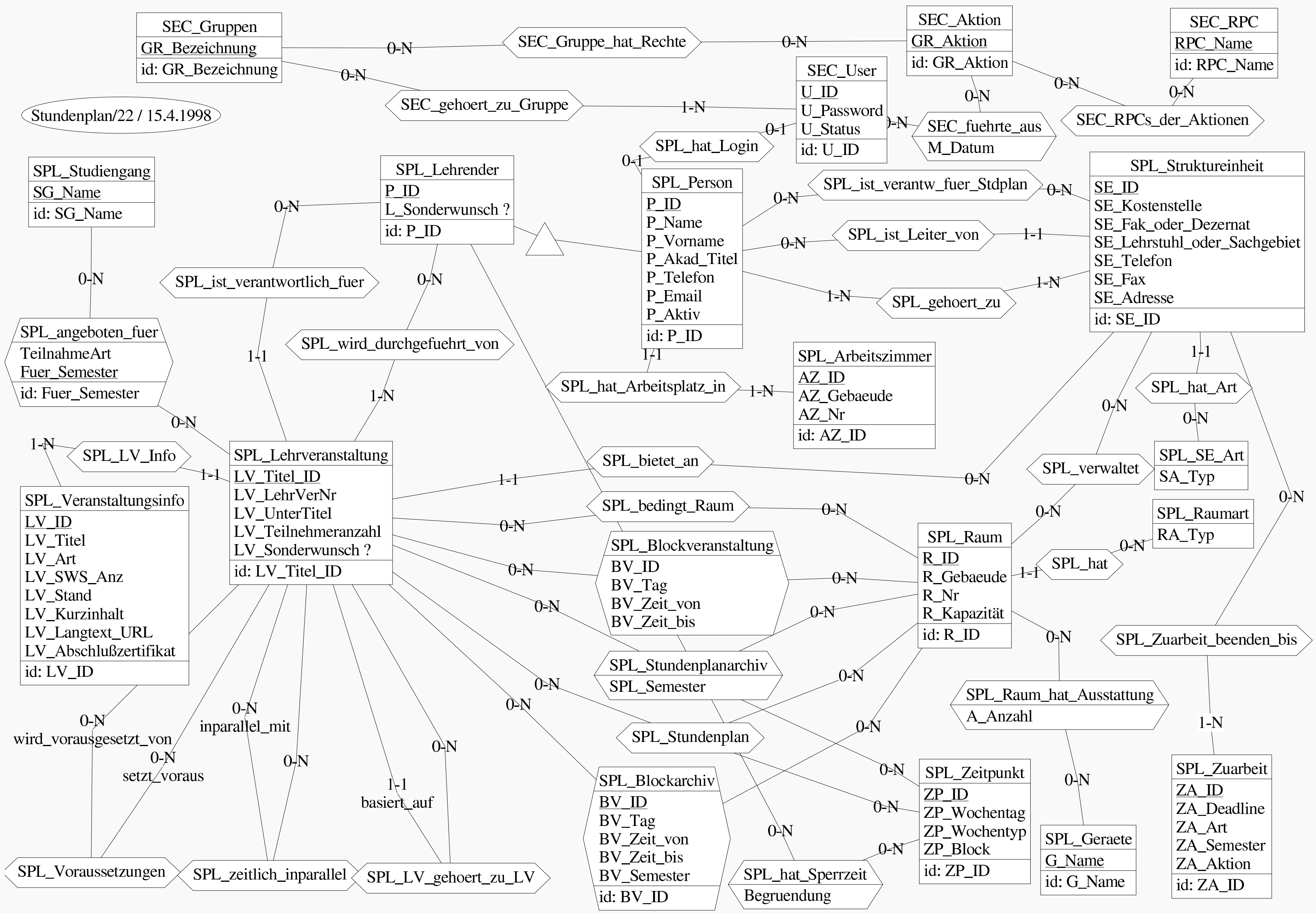
Example: Lecture Scheduling

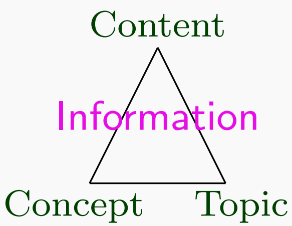
The full schema

Querying
Answering
Visuality
29.11.2013
B. Thalheim

State-of-art
Systematics
VisualSQL
Query forms
Answer forms
Question2Query
Liquefaction
Conclusion

Content
Information
Concept Topic





Example NL Analysis

Which lectures are given by Vierhaus and Thalheim?

```
s [praes]
  np [akk,plural,3,noun]
     quant [akk,pliral,fem,finit]
       welch [akk,plural,fem,finit]
     n [akk,plural,fem,finit,3,noun]
       noun [akk,plural,fem,finit,3]
         Veranstaltung [akk,plural,fem,finit,3]
  vp [[np,[nom,plural,3,noun]],plural,,3,praes]
     v [vf,[noaux,haben],[finit,nopsp],plural,3,praes,noprae]
        les [vf,[noaux,haben],[finit,nopsp],plural,3,praes,noprae]
  connp []
     np [gen,sing,3,noun]
        n [gen,sing,mas,infinit,3,noun]
           noun [gen,sing,mas,infinit,3]
              tktktk [gen,sing,mas,infinit,3]
     conn []
        und []
     np [akk,plural,mas,infinit,3,noun]
        n [akk,plural,mas,infinit,3,noun]
           noun [akk,plural,mas,infinit,3]
              tktktk [akk,plural,mas,infinit,3]
```

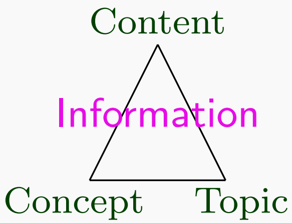
The Resulting Query

Which lectures are given by Vierhaus and Thalheim?

```
au tk 12 (.../diplom/SQL-Generator) : echo "Welche Veranstaltungen lesen Thalhei
m und Vierhaus." | ./src/syntax-analyse/sql-gen
Anfrage: Welche Veranstaltungen lesen Thalheim und Vierhaus.
 1. Pfad: Veranstaltung les Thalheim
  1. Weg: SPL_Veranstaltungsinfo -> SPL_LV_Info -> SPL_Lehrveranstaltung -> SPL_
wird_durchgefuehrt_von -> SPL_Lehrender -> SPL_Person.SPL_Lehrender -> SPL_Perso
n
  SQL-Query: select SPL_Veranstaltungsinfo.LV_ID, SPL_Veranstaltungsinfo.LV_Tite
l, SPL_Veranstaltungsinfo.LV_Art, SPL_Veranstaltungsinfo.LV_SWS_Anz, SPL_Veranst
altungsinfo.LV_Stand, SPL_Veranstaltungsinfo.LV_Kurzinhalt, SPL_Veranstaltungsin
fo.LV_Langtext_URL, SPL_Veranstaltungsinfo.LV_Abschlußzertifikat, SPL_Person.P_I
D, SPL_Person.P_Name, SPL_Person.P_Vorname, SPL_Person.P_Akad_Titel, SPL_Person.
P_Telefon, SPL_Person.P_Email, SPL_Person.P_Aktiv from SPL_Person, SPL_wird_durc
hgefuehrt_von, SPL_Lehrender, SPL_Veranstaltungsinfo, SPL_Lehrveranstaltung wher
e SPL_Lehrender.P_ID = SPL_Person.P_ID and SPL_wird_durchgefuehrt_von.P_ID = SPL
_Lehrender.P_ID and SPL_Lehrveranstaltung.LV_Titel_ID = SPL_wird_durchgefuehrt_v
on.LV_Titel_ID and SPL_Lehrveranstaltung.LV_ID = SPL_Veranstaltungsinfo.LV_ID an
d SPL_Person.P_Name = 'Thalheim'

 2. Pfad: Veranstaltung les Vierhaus
  1. Weg: SPL_Veranstaltungsinfo -> SPL_LV_Info -> SPL_Lehrveranstaltung -> SPL_
wird_durchgefuehrt_von -> SPL_Lehrender -> SPL_Person.SPL_Lehrender -> SPL_Perso
n
  SQL-Query: select SPL_Veranstaltungsinfo.LV_ID, SPL_Veranstaltungsinfo.LV_Tite
l, SPL_Veranstaltungsinfo.LV_Art, SPL_Veranstaltungsinfo.LV_SWS_Anz, SPL_Veranst
altungsinfo.LV_Stand, SPL_Veranstaltungsinfo.LV_Kurzinhalt, SPL_Veranstaltungsin
fo.LV_Langtext_URL, SPL_Veranstaltungsinfo.LV_Abschlußzertifikat, SPL_Person.P_I
D, SPL_Person.P_Name, SPL_Person.P_Vorname, SPL_Person.P_Akad_Titel, SPL_Person.
P_Telefon, SPL_Person.P_Email, SPL_Person.P_Aktiv from SPL_Person, SPL_wird_durc
hgefuehrt_von, SPL_Lehrender, SPL_Veranstaltungsinfo, SPL_Lehrveranstaltung wher
e SPL_Lehrender.P_ID = SPL_Person.P_ID and SPL_wird_durchgefuehrt_von.P_ID = SPL
_Lehrender.P_ID and SPL_Lehrveranstaltung.LV_Titel_ID = SPL_wird_durchgefuehrt_v
on.LV_Titel_ID and SPL_Lehrveranstaltung.LV_ID = SPL_Veranstaltungsinfo.LV_ID an
d SPL_Person.P_Name = 'Vierhaus'

au tk 13 (.../diplom/SQL-Generator) :
```

Plan and Achievements for this Talk

Observations for the current state-of-art

- Trapped by SQL and database schemata
- Being limited for formulation, understanding, culture

Systematic querying by reconsidering search

- Property-based search is the toughest form!
- Extension of search forms

Query forms as a framed form for query formulation

- Questions are anyway stereotyped.
- Use the stereotype for query generation.

Answer forms as a way of deriving the format of the answer

- Questions contain partially the answer format.
- Use the answer format for answer stereotypes.

Query formulation from questions

- SQL users have to state queries in the SQL form!
- Why we should not support the user?

Question liquefaction for generation of queries

- Automatic query decomposition, liquefaction and composition.
- Natural language approaches to generation.

Summarising

Systematic question transformation +

automatic query generation +

automatic answer delivery

Query formulation

as a six-step procedure

Query and answer forms

for orthonormalised questions and

for any kind of question

Tools as a proof-of-concept

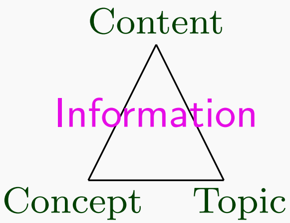
with applications in everyday life

VisualSQL

as the better form for query formulation

without the SQL burden

`http://www.informatik.uni-kiel.de/en/information-systems-engineering/miscellaneous/visualsql/`



Visualisation is not the Silver Bullet

Visualisation may mislead

Misleading comparisons: *Gravitation decreases by the square of the distance.*

Moore's, Gilder's or Metcalfe's laws without context

Metcalfe: The value of a network is proportional to the square number of nodes.

Colouring schemes, e.g., red color for *attention* in some cultural environments ...

Representation of complex structures, e.g., in medicine

Exclusive reasoning on representations, e.g., in ER diagrams

Software measures based on metrics without explicit quality criteria that have been deduced from the requirement and the environment

Simplicity of mind maps, topic maps or tree-structured ontologies, e.g., Carl von Linne's biological classification

TV, mass media, movie "information", e.g., war pictures, interpretation without background, rewritten history, physics in TV

Thank you!

thalheim@is.informatik.uni-kiel.de