Databases from a logical perspective

Ingmar Dasseville *ingmar.dasseville@cs.kuleuven.be*
Thomas Vanstrydonck *thomas.vanstrydonck@cs.kuleuven.be*
Simon Marynissen *simon.marynissen@cs.kuleuven.be*

October 31, 2017

In this exercise session we start from the concept of relational databases (in the rest of this assignment: database) and we investigate the aspects of databases from a logical perspective:

- Different logical visions for databases
- Modelling information in databases in logic
- The representation of queries on a database in logic as a modelling exercise.

1 A Database as a logical structure

Consider the next database (cf the course): From a logical perspective we

Instructor

Ray	CS230
Hec	CS230
Sue	M100
Sue	M200
Pat	CS238

Prerequ[a]

CS230	CS238
CS148	CS230
M100	M200

[a]Prerequ(a,b) means a is a prerequisite for b

Enrolled

Jill	CS230
Jack	CS230
Sam	CS230
Bill	CS230
May	CS238
Ann	CS238
Tom	M100
Ann	M100
Jill	M200
Sam	M200
Flo	M200

Grade

Sam	CS148	AAA
Bill	CS148	D
Jill	CS148	A
Jack	CS148	C
Flo	CS230	AA
May	CS230	AA
Bill	CS230	F
Ann	CS230	C
Jill	M100	B
Sam	M100	AA
Flo	M100	D
Flo	M100	B

PassGrade

AAA
AA
A
B
C

Figure 1: The student database

can directly see this database as a structure I over a vocabulary Σ.

$$\Sigma = \{Instructor/2, Grade/3, Enrolled/2, Prerequ/2, PassingGrade/1,$$
$$Ray, Hec, \ldots, CS230, \ldots, AAA, \ldots, Jill, \ldots\}$$

The domain is

$$Domain_I = \{Ray, Hec, \ldots, CS230, \ldots, AAA, \ldots, Jill, \ldots\}$$

The interpretation of I consists of

- The interpretation of all constants: $Ray^I = Ray, \ldots, CS230^I = CS230, \ldots, AAA^I = AAA, \ldots$
- The relation of tuples of domain elements $Instructor^I, Grade^I, \ldots$ as in Figure 1

I is the Herbrand structure.

Exercise 1. Formulate the next queries in Σ en evaluate them in structure I. All queries are *simple*: the answer is $true$ or $false$. On Toledo you can find an example file in which you can fill the queries and use the IDP system to answer them.

1. *Is there a prerequisite for CS238?*
   Solution:
   $$(\exists x) Prerequ(x, CS238)$$
   Answer: True.

2. *Has May passed for CS230 ?* **Solution:**
   $$(\exists x)(Grade(May, CS230, x) \wedge PassingGrade(x))$$
   Answer: True.

3. *Are all the courses which are (direct) prerequisite for M100 instructed by Ray?*
   Solution:
   $$(\forall x)(Prerequ(x, M100) \Rightarrow Instructor(Ray, x))$$
   Strange question! $M100$ has no prerequisites!
   Answer: Trivially true. He instructs each of the 0 courses.

4. *Has everyone that is enrolled in CS230 passed for at least one course instructed by Sue?*
   Solution:
   $$(\forall x)(Enrolled(x, CS230) \Rightarrow (\exists w)(\quad Instructor(Sue, w) \wedge (\exists s)(Grade(x, w, s) \wedge PassingGrade(s))))$$
   Answer: False!

5. *Did John do an exam for CS230?*
   Solution:
   $$(\exists s)Grade(John, CS230, s)$$
   Note: John is not an element of the domain of the database, a database will answer: false, but in logic the truth value of this sentence will be undefined. Note: in the IDP system you cannot write a query like this (since John is not in the vocabulary either)

6. *Are there students taking a course from every instructor?*
   Solution:

   $$(\exists x)(\forall y)((\exists w)Instructor(y, w) \Rightarrow (\exists w)(Instructor(y, w) \wedge Enrolled(x, w)))$$

   Answer: False

7. *Is there are student who is enrolled in exactly 1 course?*
   Solution:
   $$(\exists x)(\exists 1 w)Enrolled(x, w)$$
   Answer: True

Exercise 2 An *n-ary* query is a symbolic definition of a relation of the form $\{(x_1, \ldots, x_n) : A\}$. On Toledo you can find an example file in which you can write the queries and use the IDP system to answer them.
Construct the following n-ary queries:

1. *All tuples (x,y) such that student x has passed for the course y.*
   Solution:

   $$\{(x, y) : (\exists z)(Grade(x, y, z) \wedge PassingGrade(z))\}$$

   A database will answer this question with the following relation:

Sam	CS148
Jill	CS148
Jack	CS148
Flo	CS230
May	CS230
Ann	CS230
Jill	M100
Sam	M100
Flo	M100

2. *All students who follow exactly one course*
   Solution:
   $$\{x : (\exists 1 w)(Enrolled(x, w)\}$$

3. *All students who are enrolled in CS230 and have passed for all the courses they did up til now.*
   Solution:

   $\{x : Enrolled(x, CS230) \wedge (\forall y)((\exists z)(Grade(x, y, z)) \Rightarrow (\exists z)(Grade(x, y, z) \wedge PassingGrade(z)))$

Exercise 3. An essential part of the database are the integrity constraints. The database is subject to these constraints. These need to be checked when the database is changed and need to prevent the creation of invalid relations. Express the following integrity constraints in Σ (on Toledo you can find an example file in which you can fill in the constraints and call the IDP system to test if they are satisfied in the given database).

1. *No subject is a (direct) prerequisite for itself*
   Solution:

   $$\neg(\exists w) Prerequ(w, w)$$

   Satisfied.

2. *Nobody can have more than one grade for a course.*
   Solution:

   $$\neg(\exists x)(\exists w)(\exists s)(\exists t)(Grade(x, w, s) \wedge Grade(x, w, t) \wedge s \neq t)$$

   Not satisfied

3. *No student can be an instructor. (A student is someone who is enrolled in a course)*
   Solution:

   $$\neg(\exists x)((\exists w) Instructor(x, w) \wedge (\exists w) Enrolled(x, w))$$

   Satisfied In this sentence $(\exists w) Instructor(x, w)$ expresses that x is an instructor and $(\exists w) Enrolled(x, w))$ expresses that x is a student. Note that the w in those subsentences is unrelated. We have similar integrity constraints to ensure that subjects, students, instructors and grades are pairwise disjunct, so they contain no common elements.

4. *Everyone who is enrolled for a course needs to have passed all courses which are (direct) needed prerequisites for that course.*
   Solution:

   $(\forall x)(\forall w)(Enrolled(x, w) \Rightarrow (\forall t)(Prerequ(t, w) \Rightarrow (\exists s)(Grade(x, t, s) \wedge PassGrade(s))))$

   Not valid.

2 SQL

The query language SQL is a widespread language for formulating queries on database systems. SQL is based on the principles of predicate logic but has a strongly dissimilar syntax. Nevertheless it isn't difficult to see the connection to logic.

In an SQL database the vocabulary (called the *database-scheme*) is given in the form of a number of table-declaration of the form:

$$table(field_1, field_2, \ldots, field_n)$$

Here *table* is the name of the table and $field_i$ is the name of the ith column of the table. E.g. $Grade(student, course, score)$, $Enrolled(student, score)$.

Exercise 4. Given this vocabulary we can translate SQL-queries to queries in logic. Translate the next SQL query to logic (you can do this on paper instead of in IDP)

$SELECT\ student$
$FROM\ Grade\ AS\ Gr1$
$WHERE\ NOT\ EXISTS\ (\ SELECT\ score$
$FROM\ Grade\ AS\ Gr2$
$WHERE\ Gr1.student = Gr2.student\ AND$
$NOT\ (score =$ 'AA' $OR\ score =$ 'AAA'))

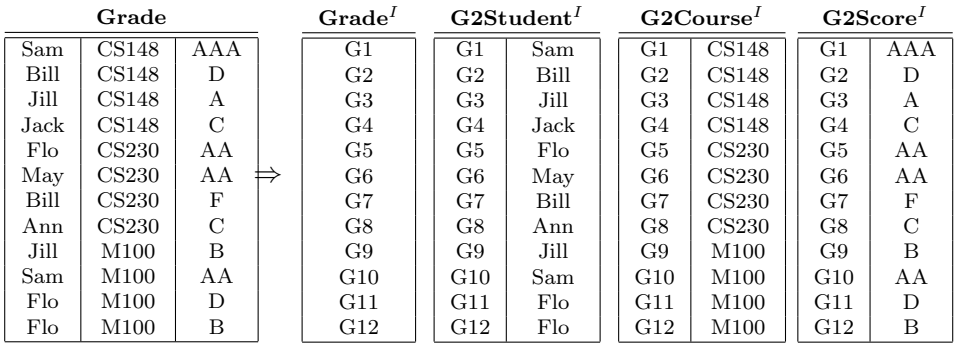
Solution:

$$\{x : (\exists y)(\exists z)Grade(x, y, z) \wedge (\forall y)(\forall z)[Grade(x, y, z) \Rightarrow z = AA \vee z = AAA]\}$$

Different database schemes can correspond to the same domain model, and even the same database scheme can be translated to different logical vocabularies. The chosen scheme has effect on the simplicity of the SQL queries, and in the same way the logical vocabulary has effect on the simplicity of modelling or execution speed of the inferences. We propose two possible ways to convert a scheme into a vocabulary.

- The "classical" way: we choose an n-ary predicate symbol per table with n columns.
- The "object oriented" way: we choose the table name *table* as a unary type-predicate and the column names as $field_i$ as binary relations. For each row of the table a new "identifier" is created who represent the objects.

The SQL-table $Grade(student, subject, grade)$ can be represented as a ternary relation as in the student database. The alternative is to view this table as a set of relations in Figure 2 for the predicate symbols $Grade/1$, $G2Student/2$, $G2Course/2$ and $G2Score/2$.



Grade		
Sam	CS148	AAA
Bill	CS148	D
Jill	CS148	A
Jack	CS148	C
Flo	CS230	AA
May	CS230	AA
Bill	CS230	F
Ann	CS230	C
Jill	M100	B
Sam	M100	AA
Flo	M100	D
Flo	M100	B

⇒

$\mathbf{Grade}^I$
G1
G2
G3
G4
G5
G6
G7
G8
G9
G10
G11
G12

$\mathbf{G2Student}^I$	
G1	Sam
G2	Bill
G3	Jill
G4	Jack
G5	Flo
G6	May
G7	Bill
G8	Ann
G9	Jill
G10	Sam
G11	Flo
G12	Flo

$\mathbf{G2Course}^I$	
G1	CS148
G2	CS148
G3	CS148
G4	CS148
G5	CS230
G6	CS230
G7	CS230
G8	CS230
G9	M100
G10	M100
G11	M100
G12	M100

$\mathbf{G2Score}^I$	
G1	AAA
G2	D
G3	A
G4	C
G5	AA
G6	AA
G7	F
G8	C
G9	B
G10	AA
G11	D
G12	B

Figure 2: A database as an alternative structure

Exercise 4 (continued). Translate the previous SQL query to this new vocabulary.

Exercise 5. Take a table for representing employees in a company: *Employee(id,surname,name,street,houseNb,city,zip,birthDate,job,salary,nbOfServedYears)*. This is a relation with 11 arguments. Suppose we want to know which employees earn more than 100.000 a year. In SQL this can be written down as:

$$\begin{array}{l} SELECT\ id \\ FROM\ Employee \\ WHERE\ Employee.salary \geq 100.000; \end{array}$$

Translate this query to logic, once with the classical vocabulary, and once with the alternative vocabulary. Which of those lies closer to the syntax of the SQL query?

Solution:

$$\{id: \ (\exists vn)(\exists n)(\exists st)(\exists stnr)(\exists g)(\exists z)(\exists geb)(\exists j)(\exists l)(\exists a) \\ (Employee(id, vn, n, st, stnr, g, z, geb, j, l, a) \wedge l > 100.000)\}$$

$$\{id : (\exists w)[Employeeid(w, id) \wedge (\exists l)(EmployeeSalary(w, l) \wedge l \geq 100.000)]\}$$

Exercise 6. Suppose the table of Employees has a primary key, consisting of the first column. Which improvements can be made to the logical vocabulary and structure? (Hint: which property do the alternative vocabulary and a primary key have in common?)

[EXTRA] Exercise 7. Suppose a database with two tables:

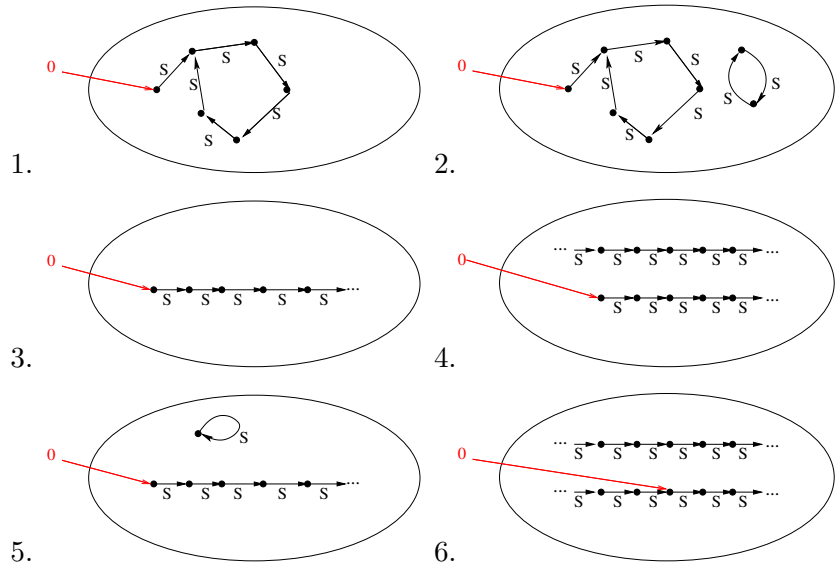
- $suppliers(supplier\_id, supplier\_name)$
- $orders(supplier\_id, quantity, price)$

The column $supplier\_id$ is a primary key in the first table.
Convert this database to a logical vocabulary and structure. Using this vocabulary, convert this query to logic.

$SELECT\ orders.quantity, orders.price$
$FROM\ orders$
$WHERE\ EXISTS(SELECT\ suppliers.supplier\_id$
$FROM\ suppliers$
$WHERE\ suppliers.supplier\_name =' IBM'$
$and\ suppliers.supplier\_id = orders.supplier\_id);$

3 UNA and DCA

Exercise 8. Here are 6 images of {0, S}-structures. In which of these images is the domain closure axiom satisfied? In which of these is the unique names axiom satisfied?



Solution:
DCA is satisfied in: 1 3
UNA is satisfied in: 3 4

Exercise 9. Define UNA(τ) and DCA(τ) for τ = { Nil : *list*, Cons: A$\times$ $list \to list$ }. Here *list* is the type of lists over A. This axiom expresses that different lists are different.

Exercise 10. Define the UNA and DCA in IDP for a type consisting of the four wind directions (North, West, South, East). This means you can not use constructed types.

Answer:

```
vocabulary V {
    type Direction
    North : Direction
    West : Direction
    South : Direction
    East : Direction
}

theory T : V {
    ! x[Direction] : x = North | x = West | x = South | x = East.
    North ~= West.
    North ~= South.
    North ~= East.
    West ~= South.
    West ~= East.
    South ~= East.
}
}
```