LIS590DCL Theory & Practice of Data Cleaning — Spring 2017
Dr. Bertram Ludäscher — March 6, 2017

Assignment 2 – Checking Data Integrity with Datalog

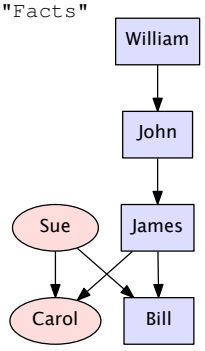
Due: March 13

Problem 1: Simple Queries in Datalog. In this assignment you will work with Datalog rules to *query* a simple database and *check* for integrity violations.[1] Consider the following EDB[2] along with (part of) the IDB[3] below. The `parent` relation is also shown as a graph, where edges point from a parent to a child:

```
% EDB (Extensional Database), "Facts"
parent(william, john).
parent(john, james).
parent(james, bill).
parent(sue, bill).
parent(james, carol).
parent(sue, carol).

male(john).
male(james).
male(bill).

female(sue).
female(carol).
```




```
% IDB (Intensional Database), "Rules"
grandparent(X,Y) :-
    parent(X, Z), parent(Z, Y).
father(X,Y)  :-
    parent(X, Y), male(X).
mother(X,Y)  :-
    parent(X, Y), female(X).
brother(X,Y) :-
    parent(P, X), parent(P, Y),
    male(X), X != Y.
sister(X,Y)  :-
    parent(P, X), parent(P, Y),
    female(X), X != Y.
```

(a) To get started, install the DLV system: `http://www.dlvsystem.com/dlv` on your computer.

(b) Read the DLV tutorial. There are some notions that will not be easy to understand at first. DON'T PANIC! Most of this will be clear by the end of this assignment. In particular try and understand the following most important subparts:

- **The Family Tree Example: Predicates, Variables, and Recursion**
- **DLV as a Deductive Database System: Comparison Operators**

We primarily need these two parts for the current assignment (read also the other parts, but don't worry if things seems rather confusing there. We'll clear up most of it in class and via a later assignment.)

(c) For the database in `family.dlv` (posted on Moodle), write rules for the following queries:

- `descendant(`X`,`Y`)` holds if X is a descendant of Y. Hint: We did `ancestor` in class.
- `sibling(`X`,`Y`)` holds if X and Y are siblings. Hint: X and Y share a parent P.

(d) ... and write the following integrity constraints (ICs) as "soft constraints" in *denial form*, i.e., write rules that yield variable bindings that serve as "constraint violation witnesses":

- (Warm-up) *Every person must have a parent*. Hint: Look at the IC-rules in the course handouts.
- (Mom & Dad) *Every person has a father and a mother.* Hint: Same idea as in the Warm-Up. First, write a rule that yields the people for which the IC is satisfied. Then write another rule that reports as an IC-violation those people who are *not* in the answer to that first query.

[1] We will also consider automatic integrity *repair* options later.
[2] Extensional Database, or *facts*.
[3] Intensional Database, or *rules*. Rules are used to define *views*, i.e., named *queries*.

Problem 2: Integrity Constraint Checking.
Consider a "dirty" dataset such as the file "`IP_publications_2015Aug28.xls`" from Assignment 1. In order to improve the data quality of the original dataset, a reasonable approach is to first apply OPENREFINE and then import the "OR-cleaned" dataset into a database. The IC-checking capabilities of a database provide a powerful way to detect inconsistencies. For this problem, assume the cleaned dataset has been loaded into a table of a relational database:

PUBLICATION

Pid	Authors	Year	Title	Journal	Vol	No	Fp	Lp	Publisher
6755	Hyatt, A.	1872	Fossil Cephalopods of the Museum of Comparative Zoology	Bull. MCZ	5	5	91	9	
2580	Rolfe, W.D.I.	1962	A new phyllocarid crustacean from the Upper Devonian of Ohio	Breviora	151	151	4	6	MCZ
2044	Francis Arthur Bather	1934	Chelonechinus N.G., A Neogene Urechinid	GSA Bull.	45	4	808	832	
4407	Kummel, B.	1969	Ammonoids of the Late Scythian	Bull. MCZ	137	3	476		
⋮									⋮

(a) Define the following ICs in *denial form* in Datalog syntax. You can assume that the table is available as a Datalog predicate of the form publication$(I, A, Y, T, J, V, N, F, L, P)$. Recall that in Datalog, arbitrary (capitalized) names can be chosen as variables, since it is the argument *position* that determines which attribute/column is meant.[4]

- (FD-1) The publication identifier Pid is a *key*, i.e., if a row agrees with another row on the key attribute Pid, then it also agrees on all other attributes (i.e., the "two" rows are in fact one and the same). As usual, your rule should return the IC-violations.
- (FD-2) Every Journal has a single Publisher. Like (FD-1), this is a *functional dependency*. It is sometimes written as Journal $\rightarrow$ Publisher.
- (NC-1) The last page Lp cannot be smaller than the first page Fp. Note: This numerical constraint can be evaluated independently on each row. Is this true for other ICs as well?

(b) Now consider that an additional table cites(P_1, P_2) is given which records pairs of publication P_1, P_2, where P_1 is citing P_2. Define the following IC in denial form:

- (ID) Every cited publication in CITES also occurs in PUBLICATION. This is an *inclusion dependency* and is usually written in the form: CITES$[\text{Pid}_2] \subseteq$ PUBLICATION[Pid].
- (NC-2) If P_1 cites P_2, then P_2's year of publication cannot be greater than P_1's year of publication.

Here is an excerpt of the CITES table:

CITES

Pid1	Pid2
4711	2020
4711	3799
3799	2580
⋮	⋮

What to submit for Assignment 2:

- 2 CR Students: **Problems 1** and **2(a)**.
- 4 CR Students: Additionally, **Problem 2(b)**.

Please submit the answers to Problems 1 and 2 in Datalog syntax. For Problem 1 also include a copy of your *execution log*, i.e., the command-line you used to solve the problem, along with the resulting output.

Further submissions details (i.e., for auto-grading) will be announced on Moodle!

[4] In contrast, SQL databases use attribute names to indicate which column is meant.