Logische Programmierung & deduktive Datenbanken — Übungsblatt 2 (Datenbank-Anfragen) —

Ihre Lösungen zu den Hausaufgaben (Aufgabe 1 bis 4) geben Sie bitte über die Übungs-Plattform in StudIP ab. Einsendeschluss ist Mittwoch, der 24. April, 16⁰⁰ (die Aufgaben werden in der Übung am Donnerstag besprochen). Aufgabe 5 wird dort auch besprochen: Sie sollten vorher darüber nachdenken, müssen aber nichts abgeben.

Hausaufgaben

Gegeben sei eine Datenbank mit Informationen über Komponisten, Stücke und CDs:

- `komponist(KNR, NAME, VORNAME, GEBOREN, GESTORBEN)`.
- `stueck(SNR, KNR→komponist, TITEL, TONART, OPUS)`.
(TONART und OPUS können NULL sein, dargestellt als '?'.)
- `cd(CDNR, NAME, HERSTELLER, ANZ_CDs, GESAMTSPIELZEIT)`.
- `aufnahme(CDNR→cd, SNR→stueck, ORCHESTER, LEITUNG)`.
(ORCHESTER und LEITUNG können NULL sein, dargestellt als '?'.)
- `solist((CDNR, SNR)→aufnahme, NAME, INSTRUMENT)`.

Sie finden die Daten unter

[<http://www.informatik.uni-halle.de/~brass/lp24/prolog/cd.pl>]

Formulieren Sie die folgenden Anfragen in Prolog, indem Sie passende Prädikate definieren, z.B. `h1` für Aufgabe 1. Wenn Ihnen bessere Namen für die Prädikate einfallen, dürfen Sie die natürlich auch verwenden. Außerdem können Sie bei Bedarf beliebige Hilfsprädikate einführen.

Aufgabe 1 (Hausaufgabe, 2 Punkte)

Geben Sie Vorname und Nachname aller Komponisten aus, die zwischen 1700 und 1799 geboren sind (jeweils einschliesslich der Grenzen).

Vergleiche werden in Prolog `<`, `=<`, `>=`, `>` geschrieben. Beachten Sie, dass `=<` (für $\leq$) anders ist als in den meisten anderen Programmiersprachen. Die Motivation dahinter ist, dass man den Pfeil `<=` vermeiden wollte.

Aufgabe 2 (Hausaufgabe, 2 Punkte)

Geben Sie alle Stücke in C-dur oder a-moll aus. Gesucht sind jeweils der Nachname des Komponisten und der Titel des Stückes.

Aufgabe 3 (Hausaufgabe, 2 Punkte)

Geben Sie Name, Vorname und Alter aller Komponisten aus, d.h. jeweils die Differenz von Todesjahr und Geburtsjahr.

Arithmetische Ausdrücke können in Prolog mit dem Prädikat `is` ausgewertet werden, z.B. `X is Y-Z`. Schreibt man dagegen `X = Y-Z`, so ist `X` anschließend ein Term wie etwa `1759-1685` für Georg Friedrich Händel.

Aufgabe 4 (Hausaufgabe, 2 Punkte)

Welcher Komponist (oder welche Komponisten) ist am „jüngsten“ in der Datenbank, d.h. hat das späteste (maximale) Geburtsjahr? Gesucht sind Name, Vorname und das Geburtsjahr.

Definieren Sie ein Hilfsprädikat für die nicht-jüngsten Komponisten und verwenden Sie dann die Negation `\+`. Beim Aufruf des negierten Rumpfliterals sollten alle Variablen in den Argumenten gebunden sein.

Zusatz

Zum Vergleich können Sie (müssen aber nicht) die Anfragen auch in SQL formulieren. Die CD-Datenbank steht auch über die Adminer-Webschnittstelle zur Verfügung:

```
[https://dbs.informatik.uni-halle.de/edb?  
pgsql=db&username=student_gast&db=postgres&ns=komponist_public
```

Das Passwort nenne ich in der Übung. Geben Sie die SQL-Anfragen nicht ab. Wir werden sie dann in der Übung besprechen.

Zur Wiederholung

Aufgabe 5 (Wiederholungs-Fragen)

- a) Schreiben Sie das Programm zur Berechnung der Vorfahren einer Person. Allgemein berechnet dies die transitive Hülle einer zweistelligen Relation, in diesem Fall der „Elternteil“-Relation.
- b) Wo wird noch eine transitive Hülle gebraucht?
- c) Kann man die transitive Hülle auch in relationaler Algebra berechnen? Wie ist die Situation bei SQL?
- d) Wie sind Formeln der Prädikatenlogik erster Stufe aufgebaut?
- e) Was legt eine Interpretation der Prädikatenlogik fest? Wann ist eine Interpretation ein Modell einer Formel oder Formelmenge?
- f) Was bedeutet es, dass eine Formelmenge F eine Formel G logisch impliziert? D.h. wie ist dieser Begriff definiert?
- g) Was bedeutet es, dass zwei Formeln F und G äquivalent sind? Nennen Sie einige Äquivalenzen.
- h) Wie kann man die Frage, ob eine Formelmenge F eine Formel G logisch impliziert auf einen Konsistenztest zurückführen? Beweiser durch Widerspruch arbeiten nach diesem Prinzip.
- i) Ist die logische Folgerung für die Prädikatenlogik erster Stufe entscheidbar?
- j) Was sind Literale (positive und negative Literale)?
- k) Definieren Sie den Begriff „Klausel“.
- l) Kann man jede Formel der Prädikatenlogik erster Stufe in eine Menge von Klauseln überführen?
- m) Was ist die Idee der Skolemisierung?
- n) Definieren Sie den Begriff Substitution. Was unterscheidet Substitutionen von Variablenbelegungen?
- o) Was sind Herbrand-Interpretationen? Was unterscheidet sie von allgemeinen Interpretationen? Warum sind sie nützlich?
- p) Erläutern Sie, wie man die Frage, ob eine Formelmenge F eine Formel G logisch impliziert auf die Existenz eines Herbrand-Modells einer Klauselmenge zurückführen kann.
- q) Skizzieren Sie kurz die Resolventenmethode zum automatischen Beweisen (auch „Resolution“ genannt). Was ist der Ableitungsschritt?

Übungsaufgaben (gemeinsam in der Übung zu bearbeiten)

Aufgabe 6 (Präsenzaufgabe)

Laden Sie sich die folgende Datei mit den klassischen Beispiel-Relationen von Oracle herunter (hier in einer etwas vereinfachten Version):

[<http://users.informatik.uni-halle.de/~brass/lp24/prolog/empdept.pl>]

Diese Datei enthält Fakten zu Abteilungen einer Firma („departments“) und Angestellten („employees“). Die Tabelle `dept` hat folgende Spalten:

- `DEPTNO` (Department Number): Nummer der Abteilung
- `DNAME` (Department Name): Name der Abteilung
- `LOC` (Location): Sitz der Abteilung.

dept		
DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

Die Tabelle `emp` hat folgende Spalten:

- `EMPNO` („employee number“): Angestellten-Nummer (Schlüssel, identifiziert Angestellten eindeutig).
- `ENAME` („employee name“): Name des Angestellten.
- `JOB`: Beruf des Angestellten.
- `MGR` („manager“): Angestellten-Nummer des direkten Vorgesetzten.
- `SAL` („salary“): Gehalt des Angestellten.
- `DEPTNO` („department number“): Abteilung, in der der Angestellte arbeitet.

emp					
EMPNO	ENAME	JOB	MGR	SAL	DEPTNO
7369	SMITH	CLERK	7902	800	20
7499	ALLEN	SALESMAN	7698	1600	30
7521	WARD	SALESMAN	7698	1250	30
7566	JONES	MANAGER	7839	2975	20
7654	MARTIN	SALESMAN	7698	1250	30
7698	BLAKE	MANAGER	7839	2850	30
7782	CLARK	MANAGER	7839	2450	10
7788	SCOTT	ANALYST	7566	3000	20
7839	KING	PRESIDENT		5000	10
7844	TURNER	SALESMAN	7698	1500	30
7876	ADAMS	CLERK	7788	1100	20
7900	JAMES	CLERK	7698	950	30
7902	FORD	ANALYST	7566	3000	20
7934	MILLER	CLERK	7782	1300	10

Formulieren Sie die folgenden Anfragen sowohl in SQL wie in Prolog. Die SQL-Version können Sie mit der Adminer-Webschnittstelle ausprobieren (Passwort in Übung):

```
[https://dbs.informatik.uni-halle.de/edb?  
pgsql=db&username=student_gast&db=postgres&ns=empdept_public
```

Bei der Lösung in Prolog erstellen Sie jeweils ein Prädikat, dessen Argumente den Spalten der verlangten Relation entsprechen.

- a) Gesucht sind Nummer und Name der Abteilung in Boston.
- b) Geben Sie Nummer und Name aller Angestellten in der Forschungsabteilung (Name: „RESEARCH“) aus.
- c) Erstellen Sie eine Liste aller Angestellten (Nummer und Name), die „MANAGER“ oder „PRESIDENT“ der Firma sind.
- d) Welche Angestellten verdienen mehr als 2500 \$?

Sie können in Prolog im Regelrumpf auch Bedingungen wie $X > Y$ verwenden. Die Variablen müssen in einer Bedingung weiter links in der Regel an einen Wert gebunden werden. Probieren Sie aus, welche Fehlermeldung Sie erhalten, wenn Sie die Rumpfliterale so umsortieren, dass mindestens eine der am Vergleich beteiligten Variablen noch keinen Wert hat, wenn der Vergleich ausgeführt wird. Beim „Constraint Logic Programming“ können Bedingungen automatisch auf einen passenden späteren Zeitpunkt verschoben werden. Prolog wertet die Bedingungen dagegen strikt in der gegebenen Reihenfolge aus.

- e) Geben Sie alle Angestellten aus, die mehr als ihr direkter Vorgesetzter verdienen. Geben Sie von Angestelltem und Vorgesetzten jeweils Name und Gehalt aus. In Prolog definieren Sie ein Prädikat `h` mit Argumenten für diese vier Werte.
- f) Welche Angestellten sind direkte oder indirekte Untergebene von „JONES“? (Für die SQL-Version können Sie voraussetzen, dass es nur zwei Hierarchie-Ebenen unterhalb von „JONES“ gibt. Sie können aber auch `WITH RECURSIVE` verwenden.)

Aufgabe 7 (Präsenzaufgabe)

Vergleichen Sie SQL und Prolog/Datalog anhand der Beispiele aus Aufgabe 6. Wenn Sie sich die Reihenfolge der Spalten von `emp` nicht merken können, wie könnten Ihnen Hilfsprädikate helfen?

Wenn Sie eine eigene logische Programmiersprache entwerfen würden, wie könnten Sie die Vorteile von Tupelkalkül (mit Spalten-Namen) und Bereichskalkül (mit Argument-Positionen) verbinden?

Zum Selbststudium

Aufgabe 8 (Zusatzaufgabe)

Schauen Sie sich die folgenden Webseiten an. Sie brauchen für diese Aufgabe nichts abzugeben. Ziel ist, dass Sie einen Eindruck davon gewinnen, was es im Internet zum Thema dieser Vorlesung gibt, und dabei für sich nützliche Quellen entdecken.

- a) „Overview of Prolog Systems“ von Ulrich Neumerkel:

[http://www.complang.tuwien.ac.at/ulrich/prolog_misc/systems.html].

Die Liste ist vermutlich ungefähr von 2004, obwohl die HTML Datei zuletzt 2015 geändert wurde.

- b) Auch die Wikipedia hat eine Liste von Prolog-Implementierungen (aktueller, aber mit etwas anderen Informationen):

[https://en.wikipedia.org/wiki/Comparison_of_Prolog_implementations].

- c) Im PC-Pool sind SWI-Prolog und GNU Prolog installiert. Da Sie SWI-Prolog schon ausprobiert haben, informieren Sie sich etwas über GNU Prolog:

[<http://www.gprolog.org/>]

Z.B. könnten Sie sich einen ersten Eindruck vom Handbuch verschaffen.

[http://www.gprolog.org/manual/html_node/index.html]

Die Benutzerschnittstelle des Interpreters wird hier erklärt:

[http://www.gprolog.org/manual/html_node/gprolog007.html#sec9]

- d) Hier zum Vergleich das Handbuch von SWI-Prolog:

[http://www.swi-prolog.org/pldoc/doc_for?object=manual]