Datenbanken, Hands-on 2

Henry Haustein

Aufgabe 1

Problem sind hier die zusammengesetzten Attribute *comments* und *categories*. In *comments* gibt es Informationen über die Seitenzahl, die Anzahl an Abbildungen und die Anzahl an Tabellen. Zudem gibt es noch weitere Informationen bezüglich der Veröffentlichung. Wie man *categories* aufspalten soll, ohne eine weitere Relation hinzuzufügen, ist mir nicht klar. Das Paper mit den meisten Kategorien hat 5 Kategorien, man könnte also Attribute wie *category1*, *category2*, *category3*, *category4* und *category5* hinzufügen. Wirklich sauber ist der Ansatz allerdings nicht und er bricht zusammen, wenn mal ein Paper mit 6 Kategorien hinzugefügt wird. Aber aus Mangel an alternativen Lösungsvorschlägen besteht dann die Relation aus den folgenden Attributen:

- *id*
- *submitter*
- *author*
- *title*
- *pages*
- *figures*
- *tables*
- *comment*
- *category1*
- *category2*
- *category3*
- *category4*
- *category5*
- *abstract*
- *update_date*

Es sind 15 Attribute.

Aufgabe 2

Die funktionalen Abhängigkeiten sind damit

- *id* $\rightarrow$ alles

- subcategory → topcategory (steht so in der Aufgabenstellung: *Jede Unterkategorie hat genau eine Überkategorie, z.B. AI kommt nur als Unterkategorie von cs vor*)

Ich habe mir noch einen Alternativaccount für OPAL organisiert und konnte so an die Musterlösung kommen. Aus dieser kann man leider nur die funktionalen Abhängigkeiten bestimmen, aber für die restlichen Aufgaben sind keine Informationen vorhanden:

- *id* → {*title*, *submitter*, *number_pages*, *number_figures*, *comments*, *abstract*, *update_date*, *author*}
- *sub_category* → *top_category*

Der einzige Schlüsselkandidat ist dann nur *id*.

Aufgabe 3

Mit diesen Abhängigkeiten kann man jetzt 3 Relationen erstellen:

- R_1: *id*, *title*, *submitter*, *pages*, *figures*, *tables*, *comment*, *abstract*, *update_date*, *author*
- R_2: *sub_category*, *top_category*
- R_3: *id*, *sub_category*

Aufgabe 4

Erstellen der Datenbanken:

```
CREATE TABLE `papers` (
  `id` VARCHAR(100) NOT NULL,
  `title` TEXT NOT NULL,
  `submitter` TEXT NOT NULL,
  `pages` INT NOT NULL,
  `figures` INT NULL,
  `tables` INT NULL,
  `comment` TEXT NULL,
  `abstract` TEXT NOT NULL,
  `update_date` DATE NOT NULL,
  `author` TEXT NOT NULL,
  PRIMARY KEY  (`id`)
) ENGINE = INNODB;
CREATE TABLE `categoryAssignment` (
  `sub_category` VARCHAR(100) NOT NULL,
  `top_category` VARCHAR(100) NOT NULL,
  PRIMARY KEY (`sub_category`)
) ENGINE = INNODB;
CREATE TABLE `categories` (
  `id` VARCHAR(100) NOT NULL,
  `sub_category` VARCHAR(100) NOT NULL,
  FOREIGN KEY (`id`) REFERENCES `papers`(`id`),
  FOREIGN KEY (`sub_category`) REFERENCES `categoryAssignment`(`
      sub_category`)
) ENGINE = INNODB;
```

Jetzt müssen die Datenbanken befüllt werden. Ich nutze hier Python um die INSERT-Befehle zu erzeugen, die ich dann vom Datenbank-Server ausführen lasse. Wir kümmern uns zuerst um die Relation R_2:

```
1  import json
2
3  # brauchen wir spaeter um Duplikate nicht in die Datenbank
       einzutragen
4  # eigentlich sollte die DB Duplikate selber erkennen, aber wir
       machen das lieber selber
5  subCategorys = []
6
7  with open("arxiv.json") as json_file:
8    # parsen der JSON-Datei
9    f = json.load(json_file)
10
11   # durchiterieren durch alle Paper in der Datei
12   for entry in f:
13     # Umwandeln der durch Leerzeichen getrennten "Liste" in ein
           Array
14     # z.B. categories = ["math.CA", "math.FA"]
15     categories = entry["categories"].split(" ")
16
17     # fuer jede Kategorie
18     for category in categories:
19       topCategory = category.split(".")[0] # z.B. "math"
20       subCategory = category.split(".")[1] # z.B. "CA"
21       if subCategory not in subCategorys:
22         # kein Duplikat
23         subCategorys.append(subCategory)
24         sql = "INSERT INTO `categoryAssignment`(`sub_category`, `
               top_category`) VALUES (\"{}\",\"{}\");".format(
               subCategory, topCategory)
25         print(sql)
```

Das liefert uns den folgenden Output:

```
1  INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("CC","cs");
2  INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("DM","cs");
3  INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("LO","cs");
4  INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("CR","cs");
5  INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("DS","cs");
6  INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("NE","cs");
7  INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("AI","cs");
8  INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("LG","cs");
```

```
9  INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("SC","cs");
10 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("CG","cs");
11 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("NI","cs");
12 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("CV","cs");
13 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("CY","cs");
14 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("DL","cs");
15 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("IR","cs");
16 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("HC","cs");
17 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("NT","math");
18 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("DC","cs");
19 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("PL","cs");
20 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("MS","cs");
21 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("RO","cs");
22 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("stat-mech","cond-mat");
23 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("data-an","physics");
24 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("CL","cs");
25 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("dis-nn","cond-mat");
26 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("CE","cs");
27 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("PF","cs");
28 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("PR","math");
29 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("GT","cs");
30 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("FL","cs");
31 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("OH","cs");
32 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("QM","q-bio");
33 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
       VALUES ("DB","cs");
34 INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
```

```
        VALUES ("MA","cs");
35  INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
        VALUES ("AO","nlin");
36  INSERT INTO `categoryAssignment`(`sub_category`, `top_category`)
        VALUES ("NA","cs");
```

Damit können wir uns jetzt um die Relation R_1 kümmern. Da die Qualität der Daten ziemlich mies ist, müssen allerhand Ersetzungen von problematischen Zeichen, wie Anführungszeichen, Slashes etc., durchgeführt werden. Die Extraktion der Anzahl an Seiten, Tabellen und Abbildungen erfolgt mit Regular Expressions (→ Einführung in die theoretische Informatik).

```
1  import json
2  import re
3
4  def getPagesFiguresTables(string):
5    # Anzahl Seiten
6    result = re.search("[0-9]+ pages", string)
7    if result:
8      pages = result.group().split(" ")[0]
9    else:
10     pages = "NULL"
11
12   # Anzahl Abbildungen
13   result = re.search("[0-9]+ figures", string)
14   if result:
15     figures = result.group().split(" ")[0]
16   else:
17     figures = "NULL"
18
19   # Anzahl Tabellen
20   result = re.search("[0-9]+ tables", string)
21   if result:
22     tables = result.group().split(" ")[0]
23   else:
24     tables = "NULL"
25
26   # den Rest des Kommentars koennte man auch noch extrahieren,
       aber ich habe keine Lust den Code dafuer zu schreiben
27   commentRest = "NULL"
28
29   return pages, figures, tables, commentRest
30
31 with open("arxiv.json") as json_file:
32   # parsen der JSON-Datei
33   f = json.load(json_file)
34
35   # durchiterieren durch alle Paper in der Datei
36   for entry in f:
37     id = entry["id"]
38     submitter = entry["submitter"]
39     title = entry["title"].strip().replace("\n", " ")
40     comments = entry["comments"]
```

```
41      # replace \ -> \\
42      # replace ' -> \'
43      # replace " -> \"
44      # replace <newline> -> <space>
45      abstract = entry["abstract"].replace('\\', '\\\\').replace("\'
            ", "\\'").replace("\"", '\\"').strip().replace("\n", " ")
46      update_date = entry["update_date"]
47      author = entry["author"]
48
49      pages, figures, tables, comment = getPagesFiguresTables(
            comments)
50
51      sql = "INSERT INTO `papers`(`id`, `title`, `submitter`, `pages
            `, `figures`, `tables`, `comment`, `abstract`, `
            update_date`, `author`) VALUES
            (\"{}\",\"{}\",\"{}\",{},{},{},{},\"{}\",\"{}\",\"{}\");".
            format(id, title, submitter, pages, figures, tables,
            comment, abstract, update_date, author)
52      print(sql)
```

Da kommen 100 SQL-Befehle raus, die ich aus Gründen der Übersichtlichkeit hier nicht hinkopiere.

Damit sind unsere Tabellen mit den Primärschlüsseln gefüllt, sodass wir nun die Relation R_3 mit Daten füllen können.

```
1  import json
2
3  with open("arxiv.json") as json_file:
4    # parsen der JSON-Datei
5    f = json.load(json_file)
6
7    # durchiterieren durch alle Paper in der Datei
8    for entry in f:
9      # Umwandeln der durch Leerzeichen getrennten "Liste" in ein
           Array
10     # z.B. categories = ["math.CA", "math.FA"]
11     id = entry["id"]
12     categories = entry["categories"].split(" ")
13
14     # fuer jede Subkategorie den Verweis setzen
15     for category in categories:
16       subCategory = category.split(".")[1] # z.B. "CA"
17       sql = "INSERT INTO `categories`(`id`, `sub_category`) VALUES
              (\"{}\",\"{}\");".format(id, subCategory)
18       print(sql)
```

Hier werden 168 SQL-Befehle erzeugt.

Damit sind alle Daten in der Datenbank und wir können endlich die Aufgaben lösen:

- Anzahl der Veröffentlichungen in Computer Science:

```
1  SELECT count(DISTINCT `categories`.`id`)
2  FROM `categories`, `categoryAssignment`
```

```
3 WHERE `categories`.`sub_category` = `categoryAssignment`.`
      sub_category`
4 AND `categoryAssignment`.`top_category` = "cs";
```

Es sind 100 (man würde auch von der Informatik-Fakultät nichts anderes erwarten, als dass der gesamte Datensatz nur Paper über Informatik enthält).

- Anzahl der Zeichen des kürzesten Autors:

```
1 SELECT `author`, LENGTH(`author`) AS `laenge`
2 FROM `papers`
3  ORDER BY `laenge` ASC;
```

Der Name ist *Fajie Li* mit insgesamt 8 Zeichen Länge, aber der Name ist nur 7 Zeichen lang.

- Seitenanzahl aller Veröffentlichungen:

```
1 SELECT SUM(`pages`)  FROM `papers`;
```

Es sind 1704 Seiten. Ein Paper hat einen Appendix, aber die Seiten von dem sollen nicht mitgezählt werden. Deswegen wurden diese Daten auch nicht mit ihn die DB aufgenommen.

- Anzahl der Abbildungen in *The Complexity of HCP in Digraps with Degree Bound Two*:

```
1 SELECT `figures`
2 FROM `papers`
3 WHERE `title` = "The Complexity of HCP in Digraps with Degree
      Bound Two";
```

Es sind 4 Abbildungen.

- Autor mit den meisten Veröffentlichungen:

```
1 SELECT `author`, COUNT(`author`) AS `anzahl`
2 FROM `papers`
3 GROUP BY `author`
4 ORDER BY `anzahl` DESC;
```

Es ist Tshilidzi Marwala mit 8 Veröffentlichungen.