Mathematisch-Naturwissenschaftliche Fakultät

Wilhelm-Schickard-Institut für Informatik

Datenbanksysteme · Prof. Dr. Grust

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Datenbanksysteme I

WS 2019/20

Torsten Grust, Christian Duta

Assignment #7

Submission Deadline: December 10, 2019 - 10:00

Please note that students will have the opportunity to **evaluate lectures**. Please help us to improve **your** courses by providing precious feedback. Check your Mailbox on **December 6th 2019**.

Exercise 1: Constraints and References (6 Points)

Answer the following questions **briefly**:

1. Explain the terms **foreign key constraint**, **inclusion constraint** and **referential integrity**.
2. What are differences between value-based references and pointers?
3. List all kinds of database constraints which were discussed in the lecture so far.

Exercise 2: Cascading Deletes (6 Points)

Create two SQL tables R and S along with their instances and constraints. Once created, all of the following properties should hold:

1. If **one** arbitrary row from `S` is deleted, it must result in **both** `R` and `S` being empty.
2. If **one** arbitrary row from `R` is deleted, it must result in `R` being empty (while `S` remains unchanged).
3. No value in a key column must be referenced by more than one value of a foreign key column.
4. Instances of both `R` and `S` must contain at least 5 rows.

Hand in a list of SQL commands that create tables, instances, and constraints with these properties. Also hand in two SQL `DELETE` statements that show that the properties 1 and 2 hold.

Exercise 3: Existential Quantification (9 Points)

Assume tables

```
CREATE TABLE r(x int);
CREATE TABLE s(x int);

ALTER TABLE r ALTER COLUMN x SET NOT NULL;
ALTER TABLE s ALTER COLUMN x SET NOT NULL;
```

Implement SQL queries to simulate the following **set operations**:

1. Set difference $R \setminus S$
2. Set intersection $R \cap S$
3. The subset relation $\subseteq$: Implement a query which outputs `true` iff $R \subseteq S$.

As these are set operators, the result should not contain duplicates. Formulate your queries using predicates `(NOT) IN` and `(NOT) EXISTS` and include test cases in your answer.

Important: Do **not** use the corresponding SQL set operators `INTERSECT`, `EXCEPT` and `UNION`.

Exercise 4: Intra-Table Foreign Keys (9 Points)

In the lecture we discussed a particular relational encoding of tree-shaped data structures (slide 12, slide set 7 "Referential Integrity"). In each node we stored a reference to its parent node.

Alternatively, we may want each node to reference its children instead. We assume that nodes in the tree have at most two children. Create a table for this alternative encoding, with appropriate constraints. Encode the tree structure from slide 12 appropriately and store it in your table.

Next, formulate SQL-queries on your tree representation to answer the following questions:

1. What is the label of the sibling of the node with label 'E'?

   result(sibling)

2. What are the labels of the grandchildren of the node with label 'A'?

   result(grandchildren)

3. What is the label of the root node?

   result(root)