TU München, Fakultät für Informatik
Lehrstuhl III: Datenbanksysteme
Prof. Dr. Thomas Neumann

Übung zur Vorlesung *Grundlagen: Datenbanken* im WS19/20

Christoph Anneser, Moritz Sichert, Lukas Vogel (gdb@in.tum.de)
https://db.in.tum.de/teaching/ws1920/grundlagen/

Blatt Nr. 01

Hausaufgabe 1

Als moderner Netzbürger besitzen Sie einen Twitter-Account mit 150 Followern. Nach der Vorlesung schreiben Sie folgenden Tweet:

a) Welche Daten werden durch das Absenden dieses Tweets generiert? An welchen Stellen im System von Twitter könnten diese gespeichert werden? Denken Sie nicht nur an den Text des Tweets sondern auch an potentielle Metadaten.

   Welche zusätzlichen Daten werden generiert, wenn Ihr Sitznachbar auf das Herz ("Like") klickt?

b) Seit dem 25.5.2018 gilt die Datenschutz-Grundverordnung (DSGVO), wegen der u.A. personenbezogene Daten auf Anfrage gelöscht werden müssen. Sie sind bei Twitter zuständig, diese Anfragen zu bearbeiten. Überlegen Sie sich, welche Daten bei so einer Anfrage gelöscht werden müssen und welche Probleme dabei auftreten können.

Lösung:

a) Natürlich wird der eigentliche Text mit Zuordnung zum Account "@netzbürger42" und dem Datum gespeichert. Da der Tweet eine Referenz auf "@DLR_de" enthält, wird eine auf diesen Tweet verweisende Benachrichtigung für das DLR erzeugt und gespeichert. Auch durch die Verwendung von Hashtags werden Metadaten erzeugt, um z.B. alle Tweets eines bestimmten Hashtags finden zu können. Alle bisher genannten Daten werden wahrscheinlich in einer relationalen Datenbank bei Twitter in verschiedenen Rechenzentren gespeichert.

   Zusätzlich werden indirekte Daten erzeugt. Zum Beispiel kann Twitter Informationen über den Benutzer, wie die IP-Adresse, GPS-Koordinaten, Browser und Betriebssystem, etc. zu statistischen Zwecken speichern. Diese werden wahrscheinlich in einer von den Tweets getrennten Datenbank verwaltet.

   Langfristig gesehen macht Twitter auch Backups, in denen die wichtigsten Daten nochmals vorhanden sind. Außerdem könnte Twitter auch Informationen zu Tweets und Hashtags weiteren Unternehmen zur Verfügung stellen, die diese dann weiter auswerten, z.B. zu Werbezwecken.

   Wenn Ihr Nachbar auf das Herz klickt, wird eine Verbindung zwischen dessen Account und Ihrem Tweet gespeichert.

b) Zunächst einmal müssen alle Tweets sowie das Nutzerkonto selbst gelöscht werden. Das umfasst Daten wie Account-Name, Password-Hashes, Anschrift, Geburtstag, etc. Da diese wahrscheinlich an einer Stelle gespeichert werden, sind sie einfach zu löschen.

Komplizierter wird es in den Fällen, wo der zu löschende Nutzer mit anderen Nutzern interagiert hat. Hier ist es wichtig, sich ein Vorgehen zu überlegen, dass die Konsistenz aller Daten garantiert. Was zum Beispiel geschieht mit Direktnachrichten? Werden nur die Nachrichten des Nutzers gelöscht, sodass der Gesprächspartner nur noch seine Antworten sieht? Oder wird das gesamte Gespräch gelöscht? Auch denkbar wäre es, die Nachrichten zu erhalten, aber den Namen durch ein Pseudonym zu ersetzen (so macht es Facebook). Dasselbe Problem existiert für Antworten auf Tweets oder Retweets.

Noch schwieriger wird es mit Daten, auf welche nicht direkt zugegriffen werden kann. Das trifft insbesondere auf Backups zu, welche oft komprimiert, oder auf Offline-Speichersystemen (Band) bzw. Read-Only-Medien (Optische Medien) gespeichert sind. Twitter kann nicht wegen jedem Löschantrag seine gesamten Backups zerstören. Twitter könnte hier z.B. Backups verschlüsseln, sodass "gelöschte" Daten nicht in die falschen Hände gelangen können sowie ihr Recovery-System so anpassen, dass es Daten von "gelöschten" Nutzern nicht wiederherstellt, sollte eine Systemwiederherstellung nötig sein.

Hausaufgabe 2

Formulieren Sie die folgenden Anfragen auf dem Universitätsschema[1] in Relationenalgebra. Geben Sie die Lösungen in der in der Vorlesung besprochenen Operatorbaum-Darstellung an.

a) Geben Sie Namen und Semester aller *Studenten* an, die mindestens im 7. Semester sind.

b) Geben Sie die Namen aller *Professoren* an, die mindestens eine *Vorlesung* mit mindestens 4 SWS lesen.

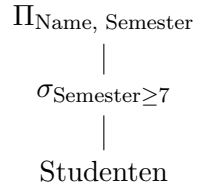
c) Geben Sie alle *Vorlesungen* an, die der *Student* Xenokrates gehört hat.

d) Geben Sie die Namen aller *Assistenten* an, deren Boss mindestens eine *Vorlesung* geprüft hat.

e) Geben Sie die Titel der direkten Voraussetzungen für die *Vorlesung* Wissenschaftstheorie an.

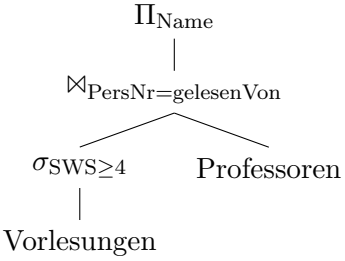
Lösung:

a) Geben Sie Namen und Semester aller *Studenten* an, die mindestens im 7. Semester sind.

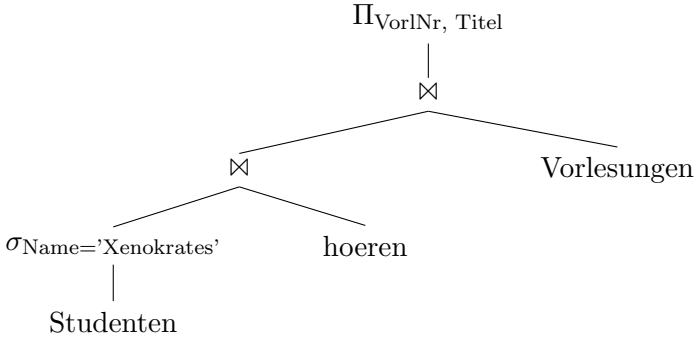


[1] `https://db.in.tum.de/teaching/ws1920/grundlagen/uni.png`

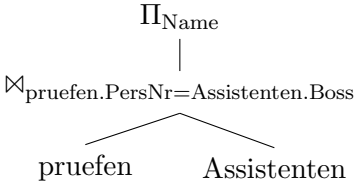
b) Geben Sie die Namen aller *Professoren* an, die mindestens eine *Vorlesung* mit minestens 4 SWS lesen.



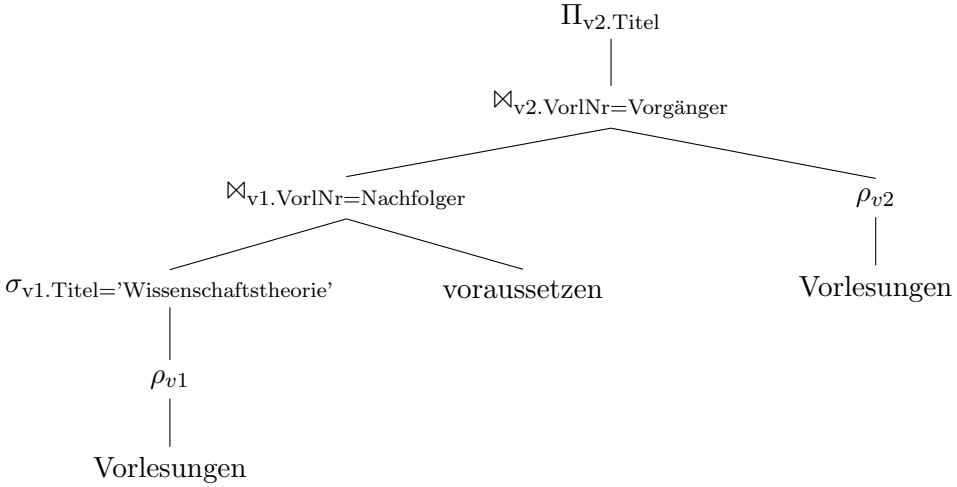
c) Geben Sie alle *Vorlesungen* an, die der *Student* Xenokrates gehört hat.



d) Geben Sie die Namen aller *Assistenten* an, deren Boss mindestens eine *Vorlesung* geprüft hat.



e) Geben Sie die Titel der direkten Voraussetzungen für die *Vorlesung* Wissenschaftstheorie an.



Gruppenaufgabe 1 (nicht Zuhause vorbereiten)

Sie designen eine Webanwendung zur Univerwaltung. Früh entschließen Sie sich zum Einsatz eines Datenbanksystems als Backend für Ihre Daten. Ihr Kollege ist skeptisch und würde die Datenverwaltung lieber selbst implementieren. Überzeugen Sie ihn von Ihrem Entschluss. Finden Sie stichhaltige Antworten auf die folgenden von Ihrem Kollegen in den Raum gestellten Äußerungen:

a) Die Installation und Wartung eines Datenbanksystems ist aufwendig, die Erstellung eines eigenen Datenformats ist straight-forward und flexibler.

b) Mehrbenutzersynchronisation wird in diesem Fall nicht benötigt.

c) Es ist unsinnig, das jeder Entwickler zunächst eine eigene Anfragesprache (SQL) lernen muss, nur um Daten aus der Datenbank zu extrahieren.

d) Redundanz ist hilfreich, wieso sollte man auf sie verzichten?

Lösung:

a) Einige Datenformate sind inhärent unflexibel, da es keinen standardisierten Pfad zur Erweiterung, Verteilung, Recovery etc. gibt. Man bedenke beispielsweise, dass allein viele Dateisysteme keine Dateien über einer fixen Größe, bei FAT32 beispielsweise traditionell 2GB erlauben. Hinzu kommt, dass durch die manuelle Erstellung von Dateiformaten keine standardisierten Datentypen verwendet werden und das gesamte "Schema" der Datenspeicherung leicht uneinheitlich wird. Im Vergleich dazu ist der Aufwand zu Erstinstallation einer Datenbank vernachlässigbar, insbesondere, da heutzutage nicht zwangsläufig ein komplexes Produkt wie IBM DB2[2] o.Ä. verwendet werden muss sondern man für kleine Eigenentwicklungen auch durchaus eine eingebettete Datenbank wie etwa sqlite[3] verwendet werden kann.

b) Mehrbenutzersynchronisation ist inhärent notwendig, wenn sie ein System entwickeln, auf das mehrere Personen zugreifen. Insbesondere ist diese eine der Eigenschaften, die nicht einfach "nachgepatched" werden kann, sondern sehr tief in eine Datenverwaltungsschicht integriert werden muss. Datenbanksysteme erlauben es, ohne über Nebenläufigkeit nachdenken zu müssen auf Daten zuzugreifen und das "erwartete" Ergebnis zu erhalten. Siehe dazu das ACID Paradigma[4], was i.A. von DBMS erfüllt wird.

c) Ein eigenes Datenformat und dessen API (wenn es denn zumindest eine API für den Zugriff gibt) muss auch gelernt werden, dafür ist SQL standardisiert und kann auch beim Wechsel des DBMS weiterverwendet werden.

d) Redundanz sorgt auch für Anomalien, etwa beim Updaten von Daten.

Gruppenaufgabe 2 (nicht Zuhause vorbereiten)

Finden Sie ein Beispiel für ein Problem (bzw. eine Inkonsistenz), die auftreten kann, wenn unkontrolliert parallel auf Daten zugegriffen wird. Ein traditionelles Beispiel hierfür ist eine gegenseitige Bank-Überweisung zwischen zwei Konten A und B. Wenn A einen Betrag x zu B überweist und B einen Betrag x' zu A, sollte immer gelten $Kontostand(A) + Kontostand(B)$ ist konstant, da sonst Geld verschwunden ist. Konstruieren Sie einen Ablauf zweier gegenseitiger Überweisungen, bei dem die Eigenschaft, dass die Kontostandssumme konstant sein soll nach dem Abschluss der zwei Überweisungen verletzt ist.

[2] `http://www-01.ibm.com/software/data/db2/`
[3] `http://www.sqlite.org/`
[4] `http://en.wikipedia.org/wiki/ACID`

Lösung:

- A liest eigenen Kontostand in Variable a ein.
- A dekrementiert a um x.
- B liest eigenen Kontostand in Variable b ein.
- B dekrementiert b um x'
- B liest As Kontostand in Variable a' ein.
- B inkrementiert a' um x'.
- B schreibt a' in As Kontostand zurück. ...