TU München, Fakultät für Informatik
Lehrstuhl III: Datenbanksysteme
Prof. Alfons Kemper, Ph.D.

Übung zur Vorlesung *Einsatz und Realisierung von Datenbanken* im SoSe25

Alice Rey, Maximilian Reif, Tobias Goetz (i3erdb@in.tum.de)
http://db.in.tum.de/teaching/ss25/impldb/

Blatt Nr. 07

Hinweise **Beachten Sie, dass dieses Blatt in KW24 und KW25 behandelt wird. Die Übungen werden zwischen 12.06. und 18.06. abgehalten. Übung 32 findet am 18.05. abweichend in 02.13.008 statt.**

Machen sie sich mit Raft vertraut: `https://thesecretlivesofdata.com/raft/`.

Hausaufgabe 1

Zeigen Sie, dass die *write-all / read-any* Methode zur Synchronisation replizierter Daten einen Spezialfall der *Quorum-Consensus*-Methode darstellt.

- Für welche Art von Workloads eignet sich dieses Verfahren besonders gut?
- Wie werden Stimmen zugeordnet um *write-all / read-any* zu simulieren?
- Wie müssen die Quoren Q_w und Q_r vergeben werden?

Hausaufgabe 2

Um Ausfallsicherheit zu garantieren, ist ein Datenwert 'A' auf vier Rechnern verteilt. Jeder Rechner hält dabei eine vollständige Kopie von 'A'. Um Konsistenz zu garantieren, wird das Quorum-Consensus-Verfahren eingesetzt. Dabei ist jedem Rechner ein Gewicht $w_i(A)$ wie folgt zugewiesen:

Rechner	Kopie	Gewicht
R_1	A_1	3
R_2	A_2	1
R_3	A_3	2
R_4	A_4	2

Das Lesequorum ist $Q_r(A) = 4$ und das Schreibquorum is $Q_w(A) = 5$.

a) Geben Sie **alle** Lesemöglichkeiten für eine Transaktion auf dem Datum 'A' nach dem Quorum-Consensus-Protokoll an.

b) Geben Sie **alle** Schreibmöglichkeiten für eine Transaktion auf dem Datum 'A' nach dem Quorum-Consensus-Protokoll an.

c) Zeigen Sie für dieses Beispiel, dass, während eine Transaktion T_1 ein Schreibquorum auf A hält, es für andere Transaktionen T_x nicht möglich ist, ein Lesequorum für A zu bekommen.

Hausaufgabe (wird nicht in der Übung besprochen) Um ein Gefühl für das *Raft Consensus Protokoll* zu bekommen, führen Sie folgende Aktionen mit *RaftScope* unter `https://raft.github.io/` aus. Die Simulation kann mit einem Klick auf das Uhr Symbol gestoppt und gestartet werden. Ein Rechtsklick auf einen der Server öffnet ein Menü um Aktionen auszulösen.

Führen Sie folgende Aktion aus und beschreiben Sie in Stichpunkten was passiert.

1. Warten sie bis die erste Leader Election abgeschlossen ist.
2. Senden Sie einen Request an den Leader. (Rechtsklick auf Leader)
3. Stoppen Sie den Leader (Server ausschalten), warten Sie bis ein neuer Leader gewählt wurde, und senden Sie einen Request an den neuen Leader.
4. Wiederholen Sie den vorigen Schritt 2x bis nur noch 2 Knoten übrig sind. Wird ein neuer Leader gewählt?
5. Starten Sie (resume) wieder einen weiteren Server. Wird ein neuer Leader gewählt?

Hausaufgabe 3

Beantworten Sie folgende Fragen zum RAFT Protokoll. Verwenden Sie *RaftScope* unter `https://raft.github.io/` um Ihre Vermutungen zu bestätigen.

1. Wie viele Server müssen ausfallen, dass in einem Cluster mit n Servern kein neuer Leader bestimmt werden kann?
2. Wie können Sie mit `restart` und `time-out` in RaftScope einen bestimmten Knoten als neuen Leader erzwingen? Geben Sie die Schritte an.
3. Wie verläuft das Beispiel in Abbildung 1 weiter? Wie kann sicher gestellt werden, dass der neu gewählte Leader den neuesten Logeintrag hält? Beschreiben Sie in Stichpunkten.

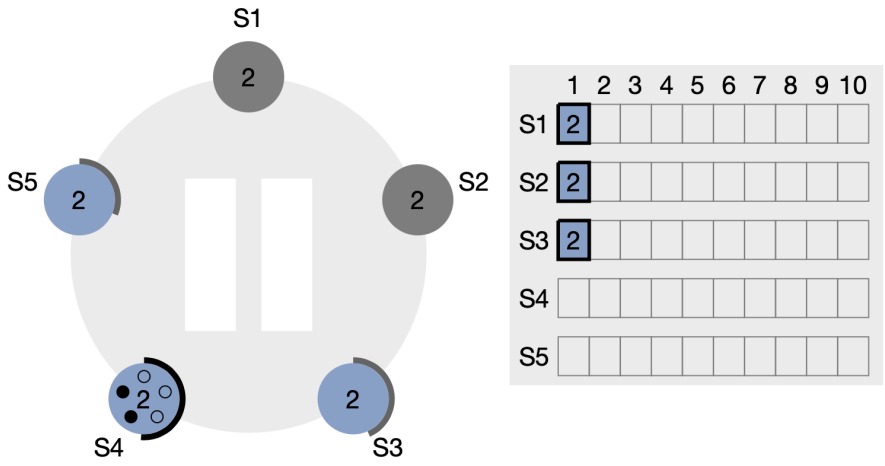

Abbildung 1: Beispiel eines Status in RaftScope

Hausaufgabe 4

Analysieren wir die Gehälter von Professoren mittels Windowfunctions und führen Sie die Abfragen unter `hyper-db.de` aus. Dazu orientieren wir uns an der Relation *Professoren* des erweiterten Universitätsschemas:

```
with Professoren (persnr, name, rang, raum, gehalt, steuerklasse) as (
  values (2125,'Sokrates','C4',226,85000,1),
  (2126,'Russel','C4',232,80000,3),
  (2127,'Kopernikus','C3',310,65000,5),
```

```
  (2128,'Aristoteles','C4',250,85000,1),
  (2133,'Popper','C3',52,68000,1),
  (2134,'Augustinus','C3',309,55000,5),
  (2136,'Curie','C4',36,95000,3),
  (2137,'Kant','C4',7,98000,1)
)
```

1. Ermitteln Sie zu jedem Professor das Durchschnittsgehalt aller Professoren.
2. Ermitteln Sie zu jedem Professor das Durchschnittsgehalt aller Professoren partitioniert nach Rang.
3. Ermitteln Sie nun die wachsende Summe (das Quantil) des Gehaltes aller Professoren partitioniert nach Rang und absteigend sortiert nach ihrem Gehalt. Gleich verdienende Professoren sind im selben Quartil.
4. Ermitteln Sie nun die wachsende Summe des Gehaltes aller Professoren partitioniert nach Rang und absteigend (total) sortiert nach ihrem Gehalt (reihenweise, nicht als Range-Query).
5. Ermitteln Sie nun das gleitende Durchschnittsgehalt aus genau zwei mehr bzw. weniger verdienenden Professoren sortiert nach Gehalt und partitioniert nach Rang.
6. Ermitteln Sie nun das gleitende Durchschnittsgehalt aus den 500 Einheiten mehr bzw. weniger verdienenden Professoren sortiert nach Gehalt und partitioniert nach Rang.
7. Ergänzen Sie zu jedem Professor das Gehalt des eins besser wie eins schlechter verdienenden.
8. Ermitteln Sie die drei bestverdienendsten Professoren einmal mit und einmal ohne Windowfunctions.

Hausaufgabe (wird nicht in der Übung besprochen) Lösen Sie folgende Anfrage mit SQL basierend auf dem bekannten Universitätsschema.

1. Bestimmen Sie die Durchschnittsnote für jeden Studenten.
2. Basierend auf dieser Durchschnittsnote, bestimmen Sie für alle Studenten ihren Rangplatz innerhalb ihrer Kohorte (Studenten desselben Semesters).
3. Berechnen Sie zusätzlich für jeden Studenten auch noch die **Abweichung** seiner Durchschnittsnote von der Durchschnittsnote der Kohorte (also vom Durchschnitt der Durchschnittsnote der Studenten der Kohorte) ausgegeben werden.

Lösen Sie Teilaufgaben 2 und 3 jeweils einmal mit und einmal ohne Nutzung von Windowfunktionen. Ihre Anfragen können Sie auf `hyper-db.de` testen. Nutzen Sie folgende erweiterte *pruefen* Relation:

```
with mehr_pruefen(MatrNr,VorlNr,PersNr,Note) as (
 select * from pruefen
 union
 values (29120,0,0,3.0),(29555,0,0,2.0),(29555,0,0,1.3),(29555,0,0,1.0)
)
```

Hausaufgabe 5

Betrachten wir das bekannte Uni-Schema mit den Faktentabellen `hoeren` und `pruefen`.

1. Ermitteln Sie in SQL mittels Fensterfunktionen (Windowfunctions) die Top-3 Studenten pro Vorlesung und geben Sie deren Namen aus.
2. Ermitteln Sie mittels SQL-92, um wieviele Notenstufen Studenten, die die Vorlesung gehört haben, in der Prüfung besser abgeschnitten haben.

Hausaufgabe (wird nicht in der Übung besprochen) Wenden wir nun Fensterfunktionen an dem fiktiven TPC-H Schema an.

1. Erstellen Sie in SQL eine Abfrage nach dem jährlichen Produktionsvolumen pro Jahr und Land. Verwenden Sie diese Abfrage als Hilfstabelle (`with`-Statement) in den folgenden Abfragen, orientieren Sie sich an TPC-H Anfrage 7.
2. Ranken Sie Länder anhand ihres jährlichen Produktionsvolumens, auf Platz eins ist das Land mit dem höchsten Volumen, das je in einem Jahr getätigt worden ist.
3. Küren Sie nun die Jahressieger. Ranken Sie dazu die Länder partitioniert nach Jahr.
4. Ermitteln Sie nun das laufende Produktionsmittel der Länge drei (Vorjahr, Nachjahr, falls erfasst).

Hausaufgabe 6

Betrachten Sie die folgende Tabelle `Waren` mit verkauften Produkten in einem Supermarkt. Die Spalte `verkauft` besagt, wieviele Einheiten des jeweiligen Produktes verkauft worden sind.

Name	Preis	Kategorie	Verkauft
Brot	1.00	Backwaren	8128
Butter	0.80	Kühlwaren	496
Grill	60.00	Haushalt	6
Steak	8.00	Kühlwaren	28
…	…	…	…

1 Ermitteln Sie in SQL mittels Fensterfunktionen (Windowfunctions) den prozentualen Umsatzanteil jedes Produktes innerhalb seiner Kategorie.

2 Ermitteln Sie in SQL mittels Fensterfunktionen (Windowfunctions) für jedes Produkt das Mittel der Verkaufszahlen aus den 5 besser verkauften (höhere Verkaufszahlen) Produkten geordnet nach Verkaufszahlen.

3 Ermitteln Sie in SQL mittels Fensterfunktionen (Windowfunctions) die drei Produkte mit dem meisten Umsatz pro Kategorie.