

Übungsblatt 9

Data-Grid und MapReduce mit Hazelcast

Aufgabe 1: Repliziertes DataGrid mit Hazelcast

In Teilaufgabe 1.1 aus Übungsblatt 8 lesen Sie innerhalb der Methode `getStudentById` Student-Objekte aus der Datenbank aus.

Nutzen Sie die Projektvorlage `uebung09_vorlage.zip` aus GRIPS oder erweitern Sie Ihr bisheriges Projekt und implementieren Sie darin folgende Funktionalität:

- Sichern Sie das neu erzeugte Student-Objekt nach Auslesen aus der Datenbank zusätzlich in einer Instanz einer `ReplicatedMap` namens `students` (Parameter bzw. Name, unter dem diese Map im Hazelcast-Data-Grid verteilt wird)
 - o Die `ReplicatedMap` soll dieses Objekt für 5 Minuten vorhalten, anschließend gilt es als veraltet (vgl. „soft state“, dies wird in der `put`-Methode individuell angegeben)
- Prüfen Sie nun zu Beginn jeder Ausführung der Methode, ob zur angeforderten Matrikelnummer bereits ein Objekt in der `ReplicatedMap` vorliegt
- Sofern ein Objekt vorliegt, geben Sie dieses zurück, eine Abfrage der Datenbank ist in diesem Fall nicht notwendig

Aufgabe 2: MapReduce

Vorbemerkungen:

Das Netbeans-Projekt aus uebung09_vorlage.zip enthält weitere Klassen, die für diese Aufgabe notwendig sind:

Im Package de.oth.vs.xml finden Sie die Klasse Veranstaltung. Diese repräsentiert eine Veranstaltung bzw. einen Veranstaltungstipp:

```
@XmlElement
@XmlAccessorType(XmlAccessType.FIELD)
public class Veranstaltung implements Serializable {
    private static final long serialVersionUID = 1L;
    private String id;
    private String titel;
    private String beschreibung;
    private Date start;
    private Date ende;
    private String owner;
    // ...
}
```

Im Package de.oth.vs.rest finden Sie die Klasse VeranstaltungResource. Hierin sind bereits zwei Methoden zum Erzeugen und Lesen einer Veranstaltung implementiert. Die Veranstaltung-Objekte werden darin in einer IMap namens "veranstaltungen" im Hazelcast-Data-Grid gespeichert bzw. daraus gelesen.

Beispielaufrufe siehe Folgeseiten!

Ihre Aufgabe:

Der Rumpf der Methode getVeranstaltungsTipps (Klasse VeranstaltungResource) ist bereits vorgegeben. Erweitern Sie diese Methode um die Funktionalität, das Data-Grid mit Hilfe des Hazelcast-MapReduce-Algorithmus nach Veranstaltungen zu durchsuchen, deren Titel oder Beschreibung mindestens eines der übergebenen Suchwörter enthalten. Das Ergebnis des Algorithmus ist eine List<Veranstaltung> mit allen Veranstaltungen, die zu den Suchwörtern passen.

Die Ergebnisliste sollte nach Relevanz sortiert werden, d. h. je mehr Suchwörter auf eine Veranstaltung zutreffen, um weiter oben in der Liste sollte diese Veranstaltung stehen. Es sollen grundsätzlich nur Veranstaltung berücksichtigt werden, die noch nicht abgeschlossen sind.

Lösen Sie diese Aufgabe mit Hilfe von MapReduce innerhalb von Hazelcast.

Implementieren Sie hierzu auch folgende Klassen im Package de.oth.vs.mapreduce:

- TippMapper implements Mapper<String, Veranstaltung, String, Veranstaltung>
- TippReducerFactory implements ReducerFactory<String, Veranstaltung, List<Veranstaltung>>
- TippCollator implements Collator<Map.Entry<String, List<Veranstaltung>>, List<Veranstaltung>>

Veranstaltungs-ID

Ein Suchwort

Ein Suchwort

Entries aus (Suchwort und Liste je Suchwort)

Gesamtliste für alle Suchwörter

Die Klasse TippMapper benötigt zusätzlich noch folgenden Konstruktor zur Übergabe der Suchwörter: public TippMapper(String[] suchwoerter)

Hinzufügen einer neuen Veranstaltung:

[-] Request

Method URL ☆ ▼ SEND

Headers Remove All

Content-Type: application/json ×

Body

```
{  
  "titel" : "34. Bayerisches Jazzweekend",  
  "beschreibung" : "Bands, Bierstände und viel Spaß bei guter (und manchmal schräger) Musik",  
  "von" : "09.07.2015",  
  "bis" : "12.07.2015",  
  "owner" : "Jazzinstitut"  
}
```

hotkey: b

[-] Response

Response Headers Response Body (Raw) Response Body (Highlight) Response Body (Preview)

7a33f311-5f35-434f-9f57-a12328fec416

Abfragen der Details zu einer Veranstaltung (über dessen kryptische UUID):

[-] Request

Method URL ☆ ▼ SEND

Headers Remove All

Accept: application/json ×

Body

Request Body

[-] Response

Response Headers Response Body (Raw) Response Body (Highlight) Response Body (Preview)

```
1. {  
2.   "id": "7a33f311-5f35-434f-9f57-a12328fec416",  
3.   "titel": "34. Bayerisches Jazzweekend",  
4.   "beschreibung": "Bands, Bierstände und viel Spaß bei guter (und manchmal schräger)  
Musik",  
5.   "owner": "Jazzinstitut"  
6. }
```

Start einer MapReduce-Anfrage für Veranstaltungstipps zu den Suchwörtern Musik und Regensburg:

<http://localhost:8080/webresources/studentisches/veranstaltung/tipps/Musik+Regensburg>

The screenshot shows a web browser interface for an HTTP client. At the top, the Method is set to GET and the URL is <http://localhost:8080/webresources/studentisches/veranstaltung/tipps/Musik+Regensburg>. A red SEND button is visible. Below the URL bar, the Headers section shows 'Accept: application/xml'. The Body section is empty. The Response section is expanded, showing the XML data. The XML is a list of events, with the first event being a benefit concert for the 'zweitesLeben e.V.' in Regensburg.

Method: GET URL: <http://localhost:8080/webresources/studentisches/veranstaltung/tipps/Musik+Regensburg> SEND

Headers: Accept: application/xml

Body: Request Body

[+] Response

Response Headers Response Body (Raw) Response Body (Highlight) Response Body (Preview)

```
--<veranstaltungen>
- <veranstaltung>
  <id>e5cce2e7-f1ca-46ee-893f-71a93338e33e</id>
  <titel>Adamar Trio</titel>
  - <beschreibung>
    Benefizkonzert für den Verein zweitesLeben e.V. mit Musik von Haydn, Mozart und Schubert im Alten Festsaal des Bezirksklinikums in Regensburg
  </beschreibung>
  <owner>MPohl</owner>
</veranstaltung>
- <veranstaltung>
  <id>7a33f311-5f35-434f-9f57-a12328fec416</id>
```