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/

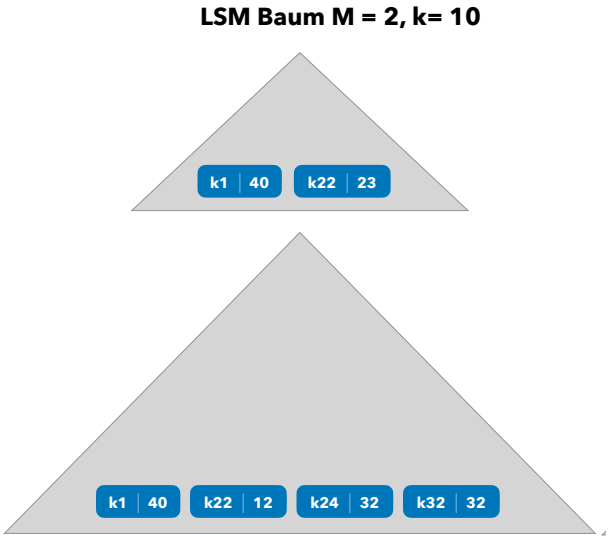
Blatt Nr. 11

Hinweise Für die aktive Teilnahme an der Vorlesung am 24. Juli benötigen Sie Spark auf Ihrem Rechner installiert. Eine Anleitung finden Sie unter `https://db.in.tum.de/teaching/ss24/impldb/Spark_preparation.pdf`. Falls Sie Schwierigkeiten haben, wenden Sie sich bitte in dieser Übungswoche an Ihren Tutor.

Hausaufgabe 1

Erklären Sie, wie LSM-Bäume das Problem der Write-Amplification beheben:

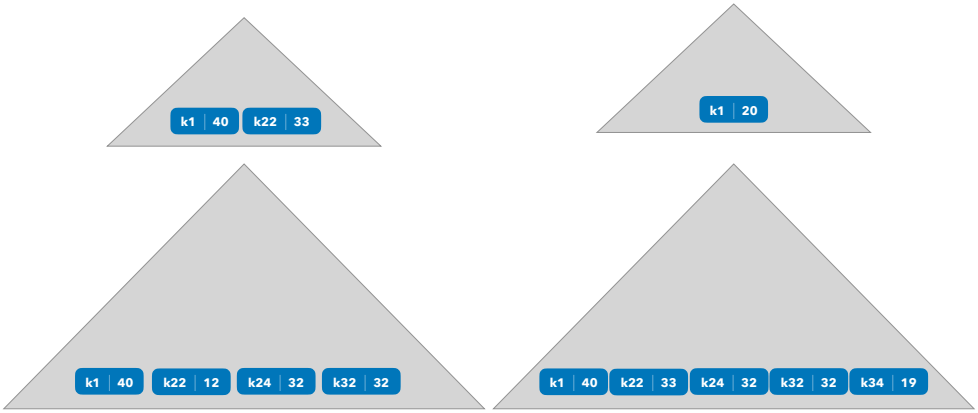
Lösung: Jeder Eintrag wird pro Level im Durchschnitt k/2 mal von unten gelesen und zurückgeschrieben bzw. gemischt. Warum k/2 mal? Intuitiv kann man sich das wie folgt klarmachen: Der allererste Eintrag wandert durch alle Level durch und wird maximal oft gemischt, nämlich k mal pro Level. Der allerletzte Eintrag wird gar nicht nach unten durchgereicht, so dass man im Durchschnitt jeden Eintrag pro Level k/2 mal lesen (für das Mischen) und schreiben (für das nachfolgende staging) muss.



- Lesen sie k22 aus dem abgebildeten LSM-Baum
- Setzen sie k22 auf 33
- Setzen sie k34 auf 19
- Setzen sie k1 auf 20

Lösung:

- Der Wert für k22 ist 23
- LSM-Baum links
- LSM-Baum rechts
- LSM-Baum rechts



Hausaufgabe 2

In dem in Abbildung 1 gezeigten Netzwerk von Web-Seiten wird ein kleines Beispiel für einen Webgraphen gezeigt. Lösen Sie folgende Aufgaben.

1. Berechnen Sie, für das in Abbildung gezeigte Netzwerk, den PageRank, sowie die HITS-Werte nach 2 Iterationen. Nutzen Sie $1/|V|$ als Anfangswert für den PageRank und 1 für HITS. $a = 0.1$
2. Formulieren sie eine Iteration des Pagerank Algorithmus in SQL. Der Graph ist dabei in der Tabelle *edges(src,dst)* gespeichert, die aktuelle PageRank Gewichtung in der Tabelle *pagerank(node,pr)*.
3. Formulieren Sie die SQL Anfrage nun als rekursive SQL Anfrage (100 Iterationen) um.

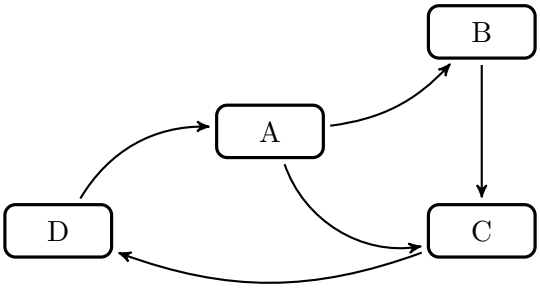


Abbildung 1: Ein kleiner Webgraph.

HITS: Iteration 1

	A	B	C	D
Hub	2	1	1	1
Auth (vorläufig)	1	2	3	1
Auth (normalisiert)	$\frac{1}{3}$	$\frac{2}{3}$	$\frac{3}{3}$	$\frac{1}{3}$

HITS: Iteration 2

	A	B	C	D
Hub	$\frac{5}{3}$	1	$\frac{1}{3}$	$\frac{1}{3}$
Auth (vorläufig)	$\frac{1}{3}$	$\frac{5}{3}$	$\frac{8}{3}$	$\frac{1}{3}$
Auth (normalisiert)	$\frac{1}{8}$	$\frac{5}{8}$	$\frac{8}{8}$	$\frac{1}{8}$

PageRank

1.

	A	B	C	D
PR Iter 1	$\frac{1}{4}$	$\frac{11}{80}$	$\frac{29}{80}$	$\frac{1}{4}$
PR Iter 2	$\frac{1}{4}$	$\frac{11}{80}$	0.2613	0.3513

2.
```
select dst, 0.1/(CAST((select count(*) from pagerank)AS FLOAT))
    +0.9*sum(Beitrag)
from(
  select e.dst, p.pr/
     (select count(*) from edges x where x.src=e.src) as Beitrag
  from edges e , pagerank p
  where e.src=p.node
) i
group by dst
```

3.
```
with recursive pagerank (iter,node,pr) as (
 select 0, e.dst, 1::float/(select count(distinct dst) from edges)
 from edges e group by e.dst
union all
 select iter+1,dst,0.1*((1::float/(select count(distinct dst) from edges)))+0.9*sum(b)
 from (
  select iter, e.dst, p.pr/(select count (*) from edges x where x.src=e.src) as b
  from edges e, pagerank p
  where e.src=p.Node and iter < 100
 ) i
 group by dst, iter
)
select * from pagerank where iter=100;
```

Hausaufgabe 3

Berechnen Sie für folgende drei Dokumente die TF-IDF-Werte:

1. „Beim Fußball dauert ein Spiel neunzig Minuten – und am Ende gewinnen die Deutschen“
2. „Beim Fußball muss das Runde (der Ball) in das Eckige (das Tor)“
3. „Nie war ein Tor so wertvoll wie jetzt“

Welches Ranking ergibt sich gemäß der Relevanzwerte für die Anfrage: „Fußball“ $\wedge$ „Tor“. Zur Ermittlung des TF Wertes gehen sie davon aus, dass alle Wörter eines Dokuments *interessant* sind?

Zur Berechnung des Rankings reicht es nur die TF-IDF-Werte von *Fußball* und *Tor* zu berechnen.

Fußball IDF: 0.176

	Dokument 1	Dokument 2	Dokument 3
TF	0.077	0.083	0
TF-IDF	0.014	0.015	0

Tor IDF: 0.176

	Dokument 1	Dokument 2	Dokument 3
TF	0	0.083	0.125
TF-IDF	0	0.015	0.022

Ranking Dokument 2: 0.029
Dokument 3: 0.022
Dokument 1: 0.014

Hausaufgabe 4

Vervollständigen Sie die untere Anfrage um die Namen der Freunde von Personen mit dem Vornamen *Sokrates* zu finden, die älter als 30 Jahre sind. Die *foaf* Onthology is unter `http://xmlns.com/foaf/spec/` beschrieben. Nutzen Sie `https://rdf.db.in.tum.de/` für Ihre Abfrage.

Lösung:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name2
WHERE {
        ?person1 foaf:knows ?person2 .
        ?person1 foaf:firstName "Sokrates" .
        ?person2 foaf:name ?name2 .
        ?person22 foaf:name ?name2 .
        ?person22 foaf:age ?age .
        FILTER ( ?age > 30 )
    }
```

Hausaufgabe 5

```
@prefix ex: <http://example.org>.
ex:Rapunzel ex:hatAutor ex:Sokrates.
ex:Rapunzel ex:erschienen 2006.
ex:Aschenputtel ex:hatAutor ex:Archimedes.
ex:Aschenputtel ex:hatAutor ex:Platon.
ex:Schneewittchen ex:hatAutor ex:Platon.
ex:Schneewittchen ex:erschienen 2004.
```

Drücken Sie die folgenden Anfragen in SPARQL aus:

1. Geben Sie alle Bücher aus, für die sowohl der Autor als auch das Erscheinungsjahr in der Datenbank enthalten sind.

```
PREFIX ex: <http://example.org>
SELECT ?book
WHERE {
   ?book  ex:erschienen ?jahr .
   ?book  ex:hatAutor ?autor .
}
```

2. Geben Sie die gemeinsamen Autoren der beiden Bücher Aschenputtel und Schneewitchen aus.

```
PREFIX ex: <http://example.org>
SELECT ?autor
WHERE {
   ex:Aschenputtel  ex:hatAutor ?autor .
   ex:Schneewittchen  ex:hatAutor ?autor .
}
```

3. Geben Sie die Namen aller Autoren (ohne Duplikate) von Büchern mit einem Erscheinungsjahr nach 2004 aus.

```
PREFIX ex: <http://example.org>
SELECT DISTINCT ?autor
WHERE {
   ?book ex:hatAutor ?autor .
   ?book ex:erschienen ?jahr.
   FILTER ( ?jahr > 2004 )
}
```

Hausaufgabe 6

wikidata.org ist ein Projekt, das strukturierte Informationen für Wikimedia-Schwesterprojekte bereitstellt. Informationen über das Datenmodell finden Sie unter https://www.mediawiki.org/wiki/Wikibase/DataModel/Primer. Praktischerweise bietet es auch eine SPARQL-Schnittstelle für Ihre Erkundungen unter query.wikidata.org.

Schreiben Sie SPARQL-Abfragen, um die folgenden Fragen zu beantworten:

1. Listen Sie alles auf, was München als Objekt verwendet. Wikidata hat den URI http://www.wikidata.org/entity/Q1726 für München vergeben. Daher können Sie unter Verwendung einer Präfixdefinition auf München verweisen, indem Sie wd:Q1726 verwenden.

```
SELECT ?a ?b
WHERE
{
       ?a ?b wd:Q1726
}
```

2. Welche Prädikat wird am häufigsten verwendet?

```
SELECT ?b (count(?a) as ?count)
WHERE
{
       ?a ?b wd:Q1726
}
group by ?b
order by desc(?count)
```

3. Welche der Städte in der Datenbank hat die früheste schriftliche Aufzeichnung?

```
SELECT ?a ?earliestRecord
WHERE
{
       ?a wdt:P31 wd:Q515;
      wdt:P1249 ?earliestRecord.
}
order by asc(?earliestRecord)
```

4. Führen Sie die Unterklassen von Sport (Q349) und ihre Bezeichnungen auf, falls es eine gibt.

```
select ?x ?name
where {
  ?x wdt:P279 wd:Q349.
  OPTIONAL {
              ?x rdfs:label ?name
              filter (lang(?name) = "en")
       }
  }
```

5. Listen Sie die transitiven Unterklassen von Sport auf.

```
select ?x ?name
where {
  ?x wdt:P279+ wd:Q349.
  OPTIONAL {
              ?x rdfs:label ?name
              filter (lang(?name) = "en")
       }
  }
```