Datenbanksysteme WS 2017/18
Prof. Dr.-Ing. Sebastian Michel
Manuel Hoffmann, Marc Dörr
TU Kaiserslautern, FB Informatik – Lehrgebiet Informationssysteme
http://dbis.informatik.uni-kl.de
Übungsblatt 12: Ausgabe 30.01.2018, Abgabe 05.02.2018 16:00 MEZ

Aufgabe 1: MapReduce (1 P.)

Wir betrachten Textdokumente, in denen jeder Satz in einer neuen Zeile gespeichert ist, etwa:

```
Dieses Buch richtet sich vor allem an Informatikstudenten der ersten Semester.
Es ist hervorgegangen aus Vorlesungen, die von den Autoren der TU Kaiserslautern gehalten wurden.
Der Inhalt und Stoffumfang wurden mehrfach in der Praxis erprobt.
...
```

Um die Sprache in diesen Texten zu verstehen, erstellen wir einen gerichteten Graphen $G = (V, E)$ wobei V aus allen Worten der Textdokumente besteht und eine Kante $(v_1, v_2) \in E$ falls in einem Satz $v_1 v_2$ direkt nacheinander vorkommen. Zum Beispiel gibt es die Knoten "der", "ersten", "TU", "Praxis" und Kanten von "der" zu "ersten", von "der" zu "TU" und von "der" zu "Praxis".

a) Geben Sie für einen MapReduce-Job Mapper und Reducer in Pseudocode an, die aus einem Dokumentkorpus den Graphen G konstruieren. Gehen Sie davon aus, dass ein Mapper jeweils eine Zeile des Textes einliest, und der Reducer mittels der Funktion `add_edge(v1, v2)` eine Kante in den Graphen einfügen kann.

b) Erweitern Sie den MapReduce-Job derart, dass Sie zu jedem Knoten zählen, wie oft das Wort vorkommt, und zu jeder Kante, wie oft das entsprechende Wort dem ersten folgt. Benutzen Sie dazu die Funktionen `add_weight(v, w)`, die dem Knoten v das Gewicht w zuordnet und `add_edge(v1, v2, w)`, die der Kante (v_1, v_2) das Gewicht w zuordnet.

Aufgabe 2: MapReduce Implementierung (1 P.)

Benutzen Sie das Java-Template von der Website, um MapReduce-Programme zu schreiben. Dabei ist wichtig, dass die Map- und Reducefunktion jeweils keine Seiteneffekte haben, sondern nur die gewünschten Werte zurückgeben.

Die Eingabe für die (erste) Mapfunktion besteht aus Key-Value-Paaren aus einer Log-Datei. Der Key ist ein Zeitstempel, der Value ein String, der den Zugriff auf ein Objekt repräsentiert. Zugriffe können sein `READ`, `WRITE` oder `DELETE`, die Objektbezeichner bestehen aus zwei Ziffern und zwischen Zugriffstyp und Objektbezeichner steht ein Leerzeichen.

a) Schreiben Sie ein Programm, das zählt, wie viele Zugriffe es von jedem Typ jeweils gab.

b) Schreiben Sie ein Programm, das alle Objekte ausgibt, auf die öfters als fünfundzwanzigmal zugegriffen wird.

c) Schreiben Sie ein Programm, das alle Objekte ausgibt, die im Abstand von 60 Minuten mindestens zweimal zugegriffen wurden. Dabei muss die Map-Funktion mehr als einen unterschiedlichen Key ausgeben.

Aufgabe 3: SQL-Anfragen in MapReduce (1 P.)

Geben Sie zu folgenden SQL-Anfragen äquivalente MapReduce-Programme in Pseudocode an.

a) `SELECT firma, AVG(gehalt) FROM pers WHERE geschlecht='w' GROUP BY firma HAVING COUNT(*)>=10`

b) `SELECT firma, name, gehalt FROM pers a WHERE NOT EXISTS (SELECT * FROM pers b WHERE b.firma = a.firma AND b.gehalt>a.gehalt)`

c) `SELECT p.name FROM pers p, firma f, chef c WHERE p.firma_id = f.id AND c.firma_id = f.id`

d) `SELECT p.gehalt FROM pers p, abteilung a1, abteilung a2 WHERE p.abt_id = a.id AND a1.id != a2.id AND a1.stadt = a2.stadt`

Können Sie diesen Join in nur einem Durchlauf von MapReduce berechnen? Falls ja, wie? Falls nicht, geben Sie eine Kaskade von MapReduce-Jobs an, die den Join berechnen. Mögliche Eingabe für Job Nr. i ist die Ausgabe vorheriger Jobs und die Basisrelationen.

Aufgabe 4: Data Layouts (1 P.)

Betrachten Sie folgende Anfragen aus dem TPC-H Benchmark.

```
select c_name, c_custkey, o_orderkey, o_orderdate, o_totalprice, sum(
   l_quantity)
from customer, orders, lineitem
   where o_orderkey in (
    select l_orderkey
    from lineitem group by l_orderkey
    having sum(l_quantity) > [QUANTITY]
)
and c_custkey = o_custkey and o_orderkey = l_orderkey
group by c_name, c_custkey, o_orderkey, o_orderdate, o_totalprice
order by o_totalprice desc, o_orderdate;
```

o_orderkey: 4 bytes, o_custkey: 4 bytes, o_orderstatus: 4 bytes, o_totalprice: 4 bytes, o_orderdate: 8 bytes, o_orderpriority: 4 bytes, o_clerk: 4 bytes, o_shippingpriority: 4 bytes, o_comment: 200 bytes. Der explizite Schlüssel pro Column habe 4 bytes.

a) Geben Sie jeweils die minimale Anzahl von Bytes an, die für Row- bzw. Column-Store für die orders-Relation von der Festplatte gelesen werden müssen.

b) Geben Sie eine geeignete vertikale Partitionierung (ohne Redundanz) an und vergleichen Sie sie bzgl. zu lesenden Bytes mit Row- und Column-Stores.

Aufgabe 5: Dies und Das (1 P.)

Dies ist eine Sammlung an Fragen, die Sie zum Teil vom Stil her auch in den Klausuren erwarten können. Andere Fragen sollen als Anregung dienen, den Stoff zu reflektieren. Natürlich sind das nicht alle Fragen, die man zu dieser Vorlesung stellen kann. Überlegen Sie sich selbst noch weitere (vgl. Blatt 2, Aufg. 5). Solche Fragen machen auch nur einen Teil der Klausur aus (vgl. Altklausur[1]).

Daneben sei noch angemerkt, dass selbstverständlich der komplette Stoff aus Vorlesung und Übung klausurrelevant ist, sofern nicht explizit herausgenommen.

1) Der Log-Ringpuffer fasst 2^{15} Einträge. Wie viele Transaktionen können gleichzeitig im Datenbanksystem ablaufen?
2) Was ist das #SAT Problem? Wo ist der Unterschied zu SAT?
3) Warum ist die extensionale Auswertung überhaupt betrachtet worden und wo liegen deren Probleme?
4) Geben Sie einen Schedule an, der in RC aber nicht in ACA liegt und beschreiben Sie was ACA bedeutet.
5) Können durch den Reduce-Side-Join Joins mit Prädikaten der Form $\theta(s,t) = |s.A - t.A| < 5$ berechnet werden?
6) Beschreiben Sie, wie innere Knoten eines R-Baums aufgebaut sind.
7) Wie können Sie durch den Reduce-Side-Join einen Full-Outer-Join berechnen?
8) Können im Replicated-Join Theta-Joins (d.h. Joins mit beliebigen Prädikaten) berechnet werden?
9) Wieso vermeidet wound-wait bzw. wait-die, dass der Wartegraph Zyklen enthält?
10) Beschreiben Sie ob Undo bzw. Redo nötig sind bei der "¬steal und ¬force".
11) Wie können Sie Skyline-Suchen mit gemischter Präferenz nach sowohl *min* als auch *max* nur unter Verwendung von *min* berechnen?
12) Was sind sichere Anfragen im Sinne probabilistischen Datenbanken?
13) Wie beurteilen Sie ein Verfahren, dass ähnlich dem R-Baum ist, aber nicht mit MB-Rechtecken sondern mit MB-Kreisen arbeitet?
14) Sie haben zweidimensionale Datenpunkte in einer Datenbank gespeichert, in der Sie keine Nächste-Nachbar-Suche durchführen können. Dafür können Sie Anfragen vom Typ "Gib mir alle Punkte in folgendem Rechteck" ausführen. Wie können Sie auf dieser Datenbank trotzdem effizient die Skyline berechnen?
15) Zeigen Sie die Echtheit der Inklusion der vorgestellten Serialisierbarkeitsklassen, CSR, VSR, FSR.
16) Diskutieren Sie, welche Änderungen an der Systemkonfiguration (VL 15, Folie 56) welche Konsequenzen zur Folge haben haben.
17) Welches der vorgestellten Hashverfahren würden Sie für einen Hashjoin benutzen?
18) Was ist der "Best-Case" für einen Performance-Vorteil von Column-Store vs. Row-Store und was ist der "Worst-Case"?
19) Was sind CLRs und warum braucht man diese?

[1] `https://kai.cs.uni-kl.de/pruefung/lv/610/`

20) Wie unterscheiden sich die folgenden Berechnungen für den Belegungsfaktor? 1. |Gespeicherte Datensätze in Primärdatei|/(|Bucketkapazität| · |Buckets in Primärdatei|) und 2. |Gespeicherte Datensätze gesamt|/(|Bucketkapazität| · |Buckets gesamt|)

21) Geben Sie einen Schedule an, in dem die Lost-Update Anomalie vorliegt.

22) Können Sie auf LSNs verzichten, solange Sie die Transaktionsnummern in den Logeinträgen speichern?

23) Warum muss man, bei der in der Vorlesung hauptsächlich betrachteten Konfiguration, auch Verlierer-Transaktionen in der REDO-Phase nachvollziehen? Und wann muss man dies nicht?

24) Wieso ist beim TPUT-Algorithmus min_k/m eine korrekte Grenze für das sequenzielle lesen?

25) Was ist der Unterschied zwischen Possible-Tuple und Possible-Answer-Set Semantik?

26) Wie ist die worst-case Komplexität beim Suchen in Quadtree und PR-Quadtree?

27) Was passiert beim zufälligen Sondieren, wenn keine freie Seite gefunden wird? Wie wird bei einer Abfrage der korrekte Datensatz gefunden?

28) Kann man die gleichen Objekte, die man in einen R-Baum einfügt, auch in einen kd-Baum einfügen?

29) Was bedeutet MinDirtyPageLSN und wo kommt dies zum Einsatz?

30) Beschreiben Sie die Idee hinter Min-Hashing und begründen Sie warum die Kollisionswahrscheinlichkeit dem Jaccard-Koeffizienten der beteiligten Mengen entspricht.

31) Welches Logging-Verfahren würden Sie benutzen, wenn a) Sie unendlich Speicher zur Verfügung haben, aber das System so instabil ist, dass häufig Recovery gemacht werden muss, oder b) wenn Recovery sehr sehr selten nötig ist, aber Sie nur langsamen und wenig Permanentspeicher verwenden können.

32) Wieso muss man bei unscharfen Sicherungspunkten beim Redo zu einem Zeitpunkt vor dem eigentlichen Sicherungspunkt beginnen?

33) Wie entsteht aus einem Konfliktschrittegraph ein Serialisierbarkeitsgraph?

34) Wann eignet sich der Replicated-Join, und wann nicht?

35) Welche Art von Index können Sie benutzen, um k-Skyband-Anfragen effizient zu beantworten?

36) Wie würden Sie die Pivot-Objekte für den GH-Tree auswählen?

37) Was ist das “Problem” mit der intensionalen Auswertung?

38) Ist ACA unbedingt erforderlich für ACID?

39) Was bedeuten DSM und NSM?

40) Was bedeuten OLTP und OLAP und wo liegen die Unterschiede?

41) Betrachten Sie das 5-Schichten-Modell. Ordnen Sie die einzelnen Themen der Vorlesung den Schichten zu.

42) Um Ergebnisse zu berechnen müssen manchmal mehrere Map-Reduce-Jobs hintereinander ausgeführt werden. Die Ausgabe des Reducers ist dabei wieder eine Eingabe für einen Mapper.

43) Was ist Late-Materialization?

44) Wann ist eine Aggregationsfunktion (im Kontext von Top-K Algorithmen) monoton?

45) Was ist für leseintensive Workloads bei Linarem Hashing besser: kontrollierte Splitting mit niedrigem β_s, mit hohem β_s oder unkontrolliertes Splitting?

46) Können TI-Datenbanken in PC-Datenbanken ausgedrückt werden, falls ja wieso?

47) Wieso gibt es beim R-Baum den Parameter m, mit dem man fordert, dass jeder Knoten mindestens $m \leq \lceil M/2 \rceil$ Einträge hat?

48) Geben Sie ein Beispiel einer BID-Datenbank an. Wofür steht BID?

49) Beschreiben Sie eine Kaskade von Map-Reduce-Jobs, die als Eingabe einen ungerichteten Graphen als Adjazenzliste nimmt, und als Ausgabe die Anzahle der Dreiecke in diesem Graphen (das sind drei Knoten, die alle miteinander verbunden sind) ausgibt.

50) Beschreiben Sie eine Möglichkeit, den kd-Baum so zu erweitern, dass mehrere Datenpunkte pro Knoten abgespeichert werden können. Erklären Sie, wie dann eingefügt, gesucht und gelöscht wird.

51) Versuchen Sie das Prinzip der MINDIST- und MINMAXDIST-Abschätzungen im R-Baum auf andere Szenarien zu übertragen.

52) Erweitern Sie das in der Vorlesung vorgestellte Zeitstempelverfahren um mehrere Versionen eines Datensatzes. Dabei werden Leseoperationen erweitert, sodass mit $r(x, t)$ angegeben wird, dass das Objekt x im Zustand vom Zeitpunkt t gelesen wird. Wie sieht ein Schreibzugriff aus? Wie viele Versionen eines Objektes müssen gespeichert sein? Wird so noch ACID garantiert?

53) Wieso gibt es verschiedene Isolationsstufen, wenn doch durch I in ACID eigentlich volle Serialisierbarkeit verlangt wird?

54) Finden Sie ein Beispiel, das zeigt, dass VSR nicht monoton ist.

55) Betrachten Sie die folgende Operation: $w_1(x)$, wobei x eine Zeile einer Tabelle ist. Was muss man beachten, um Mehrbenutzeranomalien zu verhindern; Stichwort: Indexe und materialisierte Sichten?

56) Was bedeutet OCSR und wo liegt der Unterschied zur Klasse CSR?

57) Wieso ist der Replicated-Join ein Map-Only-Join?

58) Kann man bei einem Quadtree die Himmelsrichtungen N, S, E und W hinzufügen, um einen höheren Fanout zu bekommen?

59) Vergleichen Sie Quadtree, kd-Tree und Z-Kurve in Hinblick auf zweidimensionale Bereichssuchen (also "Welche Objekte liegen in diesem Rechteck?").

60) Wieso ist der 1-Bucket-Random Algorithmus korrekt und was bedeutet dies im Sinne von Theta-Joins?

61) Wie relevant sind die Kapitel über Join-Implementierung und -Optimierung, wenn man annimmt, dass alle Basisrelationen im Hauptspeicher vorliegen?

62) Ist es beim Reduce-Side-Join im Reduce-Task notwendig die Join Bedingung zu überprüfen?

63) Geben Sie ein Beispiel mit zwei Relationen an mit einem korrekten und einem inkorrekten Plan bzgl. extensionaler Auswertung. Wo liegt das Problem?

64) Geben Sie ein Beispiel an, was passieren kann, wenn das DBMS kein WAL benutzt bzw. die Commit-Regel nicht befolgt.

65) Beschreiben Sie den Unterschied zwischen wound-wait und wait-die für eine Transaktion t_i die mit Transaktion t_j in Konflikt gerät und t_i vor t_j gestartet wurde.

66) Während der Ausführung einer Transaktion stürzt das Datenbanksystem ab. Welche der "Buchstaben" aus ACID kommt nun zum Tragen?

67) Beschreiben Sie, wie Sie Einfügen und Löschen für den GH-Tree implementieren würden.

68) Beschreiben Sie wie im Timestamp-Ordering ein Konflikt erkannt wird.

69) Beschreiben Sie, wie innere Knoten eines M-Baums aufgebaut sind.

70) Was ist die Idee hinter PAX?

71) Wie verhalten Sich statische Hashverfahren, wenn die Seitengröße beliebig groß oder klein wird?

72) Welches Layout ist für OLTP (in der Regel) besser geeignet, Row- oder Column-Store?

73) Widerlegen oder begründen Sie folgende Aussage. Im 1-Bucket-Random Algorithmus wird Tupel $s \in S$ und Tupel $t \in T$ auf keinen Fall zwei mal gejoined.

74) Beschreiben Sie die beiden Phasen von 2PL und geben Sie einen Schedule an, der nicht in Gen(2PL) liegt.

75) Was bedeutet Recoverable? Geben Sie eine Historie an, die in RC liegt und eine, die nicht in RC liegt (aber in CSR).

76) Vergleichen Sie die Hashverfahren mit Cachingverfahren, die Sie aus Rechnersysteme o.ä. Vorlesungen kennen. Was sind Gemeinsamkeiten und Unterschiede der Funktionsweisen und Anforderungen?

77) Wo liegt der Unterschied zwischen SS2PL, S2PL und 2PL? Welcher Scheduler ist restriktiver?

78) Beschreiben Sie Vor- und Nachteile von Kompression in Column-Stores?

79) Warum wird in der Regel aus Performancegründen die Konfiguration "steal und ¬force" betrachtet?

80) Warum sind Row-Stores nicht (gut) geeignet, um Kompression anzuwenden?

81) Beschreiben Sie ein Deadlock-Vermeidungsverfahren, bei dem jeder Transaktion eine Priorität $\in \mathbb{N}$ zugeordnet ist und indem bei der Vermeidung eines Deadlocks die Summe der Prioritäten der neugestarteten TA minimiert werden soll.

82) Wieso ist es nützlich zu wissen, wie kompliziert es ist, materialisierte Sichten zu warten?

83) Geben Sie eine Möglichkeit an, wie Daten aus einem höherdimensionalen Raum auf einen 1-dimensionalen Raum abgebildet werden können.

84) Beschreiben Sie was Aktionskonsistenz bzw. Transaktionskonsistenz bedeutet und wo dies für welche Checkpoint-Verfahren betrachtet wird.

85) Wie können Sie durch den Replicated-Join einen Semi-Join berechnen?

86) Geben Sie Map- und Reduce-Funktionen an, um einen invertierten Index aufzubauen.

87) Können Sie für jeden mehrdimensionalen Baum einen Fall finden, in dem Sie diesen einem der anderen Bäume vorziehen würden?

88) Was geht beim FA Algorithmus schief, wenn die Aggregationsfunktion nicht monoton ist?

89) Geben Sie für die beiden Priorisierungen MINDIST und MINMAXDIST je einen Fall an, in der die eine Strategie der anderen deutlich überlegen ist. Können Sie eine Faustregel angeben, wann man welche Strategie benutzen soll?

90) Beschreiben Sie für die einzelnen in der Vorlesung betrachteten Aspekte, welchen Einfluss deren Fehlen auf ein DBMS hat.

91) Wieso sind Redo und Undo-Phase idempotent und was bedeutet dies überhaupt?

92) Wieso ist es wünschenswert, eine monotone Serialisierbarkeitsklasse zu benutzen?

93) Beschreiben Sie Unterschiede zwischen transaktionskonsistentem und unscharfem Checkpointverfahren bzw. Performance zum Anlegezeitpunkt bzw. des Wiederanlaufs.

94) Beschreiben Sie, wie sich das Vorhandensein von sekundären Indexen auf die Transaktionsverwaltung und die möglichen Anfragen auswirkt.

95) Welchen Konflikt umgeht Thomas' Write Rule?

96) Wieso ist es wichtig, dass vor einem Commit alle Informationen, die notwendig sind, um die TA nachzuvollziehen, auf persistenten Speicher geschrieben werden? Welchen "Buchstaben" aus ACID betrifft dies?

97) Ist der Threshold Algorithmus nie schlechter als Fagins Algorithmus? Welches Maß für "besser" bzw. "schlechter" benutzen Sie?

98) Ist RC unbedingt erforderlich für ACID? Und falls ja, um welchen Teil von ACID geht es dabei?

99) Geben Sie für die drei Pruning-Strategien für die Nächste-Nachbar-Suche im R-Baum jeweils einen Fall an, in dem die Strategie sehr nützlich, und einen, in dem die Strategie nicht nützlich ist.

100) Im R-Baum müssen die MBRs aller Kindknoten komplett im MBR des Elternknotens liegen. Könnte man diese Forderung auch abschwächen und sagen, die Kind-MBRs müssen das Eltern-MBR nur schneiden?

101) Beschreiben Sie anhand der Illustrationen im Script genau, wie die Suche nach einem Schlüssel bei Erweiterbarem Hashing funktioniert.

102) Wundern Sie sich, wieso nirgendwo steht, wie Sie den Punkt für diese Aufgabe erhalten? Dafür, dass sie jetzt die Fragen zumindest überflogen haben, bekommen Sie den geschenkt. Viel Erfolg bei der Prüfungsvorbereitung!