1 Allgemeine Informationen

What's this

Dies ist eine studentische, prüfungsschwierigkeit immitierende Aufgabenzusammenfassung die Themen der Vorlesung IDB. Es handelt sich insbesondere nicht um einen sogenannten "Braindump". Die Aufgaben orientieren sich an den Aufgaben in der ursprünglichen Klausur insofern, dass hier zum Beipsiel anstatt der Frage 'Sind Kühe Lila?' die Frage 'Machen Kuehe Schokolade?' steht. Auf beide Fragen ist die Antwort: Nein, du schaust zuviel Milkawerbung.

Maintaining

Der Latex-Source dieses PDFs wird auf *https://gitlab.cs.fau.de/ik15ydit/latexandmore* (ein Account ist noetig) maintain't, es wurde aber urspruenglich von einer anderen Person erstellt, die jedoch anonym bleiben moechte. Da die Aufgaben alle neu formuliert und konzipiert wurden, kann es sein, dass uns an einigen Stellen Fehler Unterlaufen sind. Solltet ihr solche Fehler finden oder generell Anmerkungen haben koennt ihr mit einem Account auf *gitlab.cs.fau.de* eine Issue aufmachen oder einen Pullrequest stellen.

Aufgabe 1: Ankreuzen

a) Was gilt in einer sequentielle Satzdatei?

- Jeder Satz in der Datei kann verkleinert werden
- Es koennen Wahlfrei Saetze aus der Datei gelesen werden.
- Die Datei kann um einen Satz am Ende erweitert werden
- Jeder Satz in der Datei kann vergroessert werden
- Die Datei kann als ganzes ueberschrieben werden
- An jeder Stelle in der Datei kann ein Satz eingefuegt werden

b) Was gilt fuer Bloecke, Seiten und Saetze

- Blöcke können kleiner als Sätze sein
- Seiten sind immer größer als Sätze
- Sätze können kleiner als Blöcke sein
- Sätze sind immer kleiner als Seiten
- Seiten sind immer größer als Blöcke
- Blöcke koennen größer als Seiten sein

c) Was ist der Systemkatalog, wofuer wird er verwendet. Nenne mindestens drei Beipiele was dort normalerweise gespeichert wird.

d) Wieviele Indirektionen koennen beim Zugriff auf einen Satz ueber seine TID maximal auftreten, wenn der Satz komplett in einer Seite gespeichert ist?

e) Welches Problem kann beim folgenden Vorgehen auftreten?

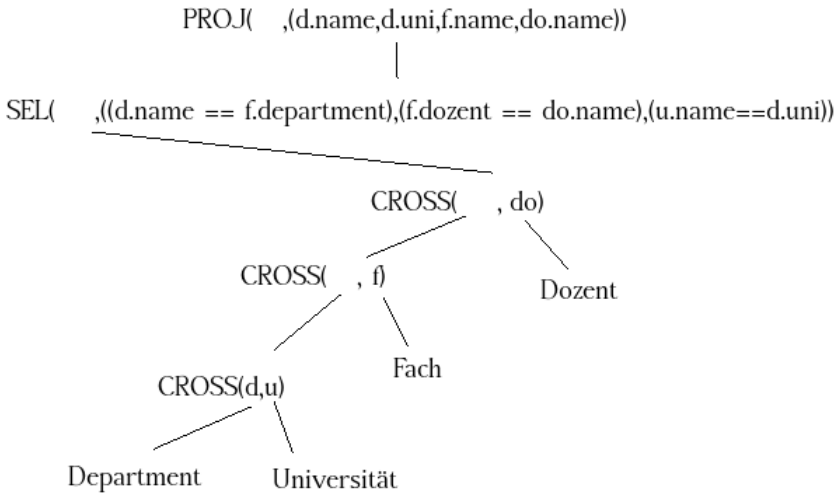
In einder Datenbank wird eine Sperre fuer ein Datenobjekt angefordert sobald das erste mal auf dieses Datenobjekt zugegeriffen wird und wieder freigegeben, sobald dieses Datenobjekt nicht mehr genutzt wird.

Aufgabe 2: SQL etc.

a) Erklaeren sie kurz den Begriff "Grenztrefferrate".

b) Was sind Planoperatoren und wie werden sie ausgewaehlt?

c) Optimiere folgenden Operatorengraph, Zwischenschritte geben keine Extrapunkte.



d) Schauen sie sich diese zwei SQL-Codeabschnitte an:

```
CREATE VIEW Bierampel AS
SELECT x.beerrating, y.winerating z.year
FROM Rel_1 x JOIN Rel_2 y ON x.ID = y.ID
WHERE x.selector = 42;
```

```
SELECT year, AVG(winerating)
FROM Bierampel
GROUP BY year
HAVING COUNT(*) > 4;
```

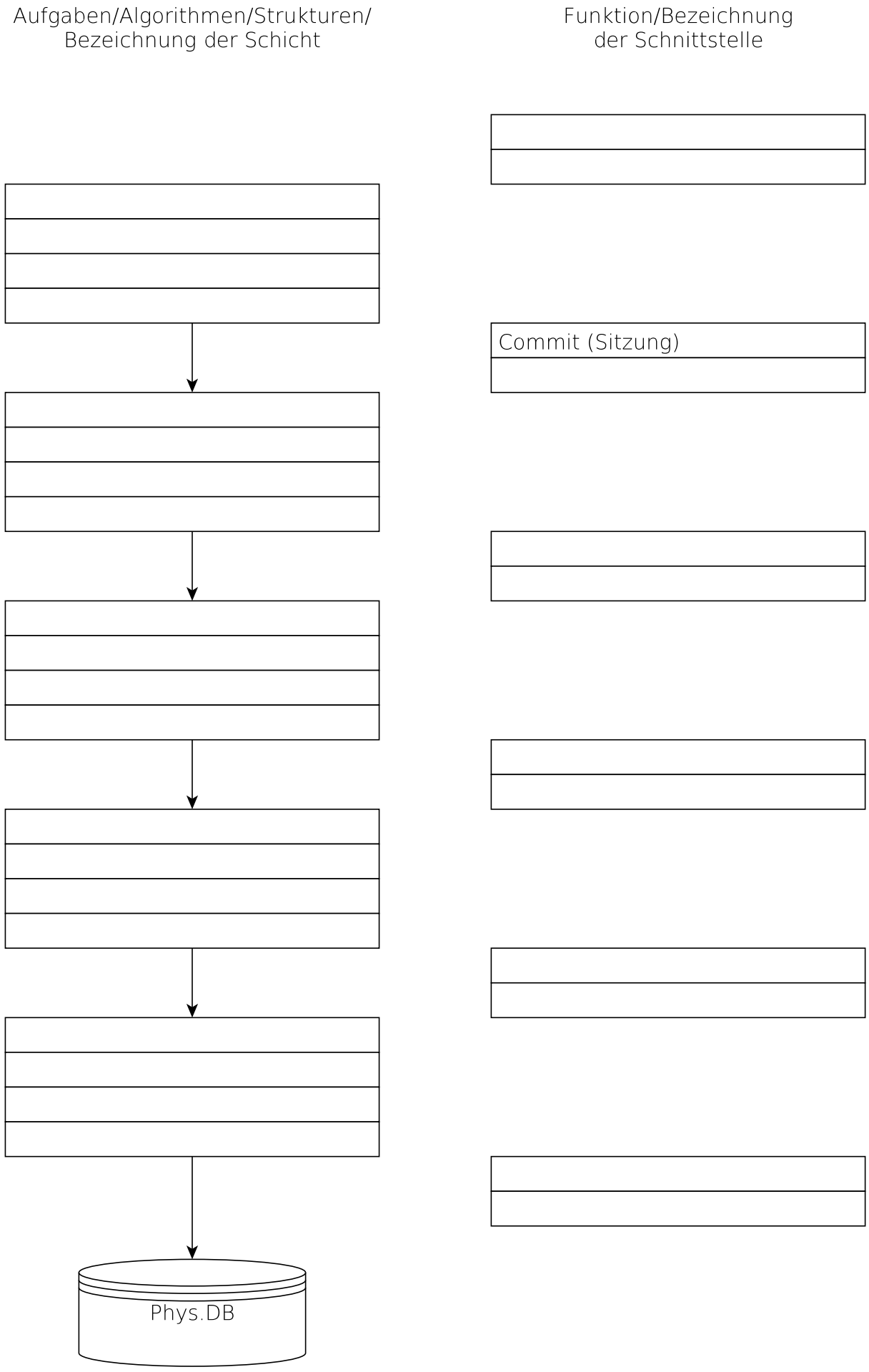
i) Zeichne einen nicht optimierten Operatorengraphen

ii) Zeichnen sie den optimierten Operatorengraphen. (auch die View muss optimiert sein)

Aufgabe 3: Schichtenmodell

Ordnen Sie ein, falsche Einordnung gibt Punktabzug:

- Speicherstrukturen
- Optimierung von SQL-Anfrage
- append(Datei, Blockanzahl)
- Freispeicherliste
- write(TID)
- Kanalkommandos
- B-Baum
- Schattenspeicher
- fix(File,BlockNo,mode)
- Ausführungsplanerstellung
- Lineares-Hashing
- mengenorientierte DB-Schnittstelle


Aufgaben/Algorithmen/Strukturen/
Bezeichnung der Schicht
Funktion/Bezeichnung
der Schnittstelle
Commit (Sitzung)
Phys.DB

Aufgabe 4: Puffer

a)

Aus der Vorlesung sind die Funktionen der Pufferschnittstelle bekannt. Ergaenzen sie den folgenden Programmablauf mit ensprechenden Funktionsaufrufen aus dieser Scchnittstelle. Fehlerbehandlung sowie Behandlung von fragmentierten Saetzen ist nicht notwendig.

```
static void read(char* tid, uint_32 index){

    char* page_in_main_memory :=

    if (indirection(page_in_main_memory, tid.index)){

        char* tid_indirected = getIndirection(page_in_main_memory, index);

        page_in_main_memory =

        tid = tid_indirected;

    }

    char* final := special_memcpy(page_in_main_memory, index);

    return satz;
}
```

b)

Bestimme die LRU-Stacktiefenverteilung fuer die folgende Zugriffsfolge. Vorgehen muss nachvollziehbar sein, markieren sie das Endergebniss gut sichtbar.
Zugriffsfolge: 4, 2, 2, 1, 4, 1, 1, 2, 4

Aufgabe 5: Koksing

Es gilt die gleiche Hashfunktionen $h_j(k) = k \bmod(2^j)$; $j = 0, 1, ...$ wie in der Vorlesung und die anfangs Bucketanzahl $q = 2$. Bucketgroesse b ist 2, geplittet wird immer wenn ein Wert in einen Overflow-Bucket geschrieben werden muss. Gewuenscht ist nur das Endergebnis, die Teilaufgaben sind voneinander unabhaenig.

a)

In die folgende Hashtabelle soll die Zahl 32 eingefuegt werden. Hashfunktionen die gerade benutzt werden:

- vor Einfügen: $h_0(k), h_1(k)$
- nach Einfügen: _____

Pointer		p						
Bucket	0	1	2	3	4	5	6	7
Feld 1	16							
Feld 2	10							

Overflowbuckets:

Feld 1								
Feld 2								

b)

In die folgende Hashtabelle soll die Zahl 36 eingefuegt werden. Hashfunktionen die gerade benutzt werden:

- vor Einfügen: $h_0(k)$
- nach Einfügen: _____

Pointer	p							
Bucket	0	1	2	3	4	5	6	7
Feld 1	12	9						
Feld 2	36	1						

Overflowbuckers:

Feld 1								
Feld 2								

c) [4 Punkte]

In die folgende Hashtabelle soll die Zahl 20 eingefuegt und danach die Zahl 7 werden. Hashfunktionen die gerade benutzt werden:

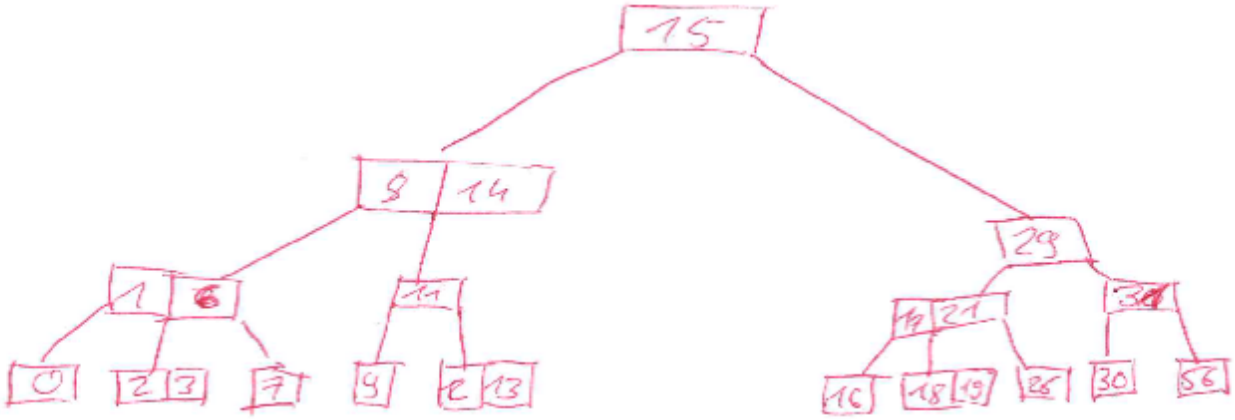
- vor Einfügen: $h_1(k), h_2(k)$
- nach Einfügen: _____

Pointer			v					
Bucket	0	1	2	3	4	5	6	7
Feld 1		3	4	14				
Feld 2		11		43				

Overflowbuckets:

Feld 1								
Feld 2								

Aufgabe 6: B-Baum



a)

Füge den Schlüssel 5 ein und zeichne den Baum.

b)

Füge den Schlüssel 25 ein (in den urspruenglichen Baum).

c)

Löschen sie die Wurzel des urspruenglichen Baumes.

Aufgabe 7: C-Store

a) Wie werden Attributswerte in C-Store gespeichert? (Kurzer Umriss)

b) Anhand welcher Eigenschaften von Attributen werden Komprimierungsverfahren in C-Store ausgewaehlt?

c) Wieviele Attribute, aus wie vielen verschiedenen logischen Tabellen, kann eine C-Store-Projektion maximal enthalten? (Keine Begruendung)

d) Wie koennen Zeichenketten extrem Platzsparend gespeichert werden?

e) In wie vielen C-Store Projektionen kann ein Attribut maximal enthalten sein? (Keine Begruendung)

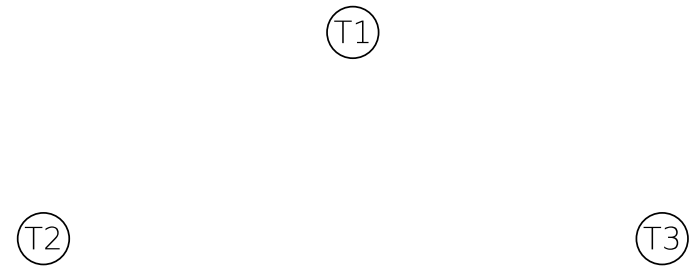
f) Kreuze *nicht* korrekte Aussagen an: (0-5)

- der Storage Key wird fuer alle Attibute mitgespeichert
- der Storage Key ist fuer alle Attributswerde eines Tupels gleich
- der Storage Key laesst sich aus der Speicherposition des Attibutwertes berechnen
- ein Verbund-Index ordnet einen Storage Key einer Projektion den einer anderen Projektion zu
- mindestes einer der Schluessel einer Projektion ist in allen anderen Tabellen enthalten

Aufgabe 8: Keiner will Backup, alle wollen Restore

a) Zeichnen sie einen Abhaengigkeitsgraphen zum folgenden Ablauf. Kanten muessen mit dem/den Objekt(en) durch das/die sie entstanden sind (also A,B oder C) gekennzeichnet werden.

w2(B), r1(C), w3(C), r1(A), w1(A), w2(B), r3(B), r3(A), w3(A), w2(B)



b) Ist der Graph oben serialisierbar? Warum bzw. Warum nicht?

d) Was gilt bei Transaction-Oriented Checkpoints? (multiple choice)

- Redo-Recovery nötig, aber durch Checkpoint begrenzt.
- Undo-Recovery nötig, aber durch Checkpoint begrenzt.
- Keinerlei Redo-Recovery nötig.
- Keinerlei Undo-Recovery nötig.
- Redo-Recovery nicht durch Checkpoint begrenzt.
- Undo-Recovery nicht durch Checkpoint begrenzt.

c) Was gilt bei Action-Consistent Checkpoints (multiple choice)

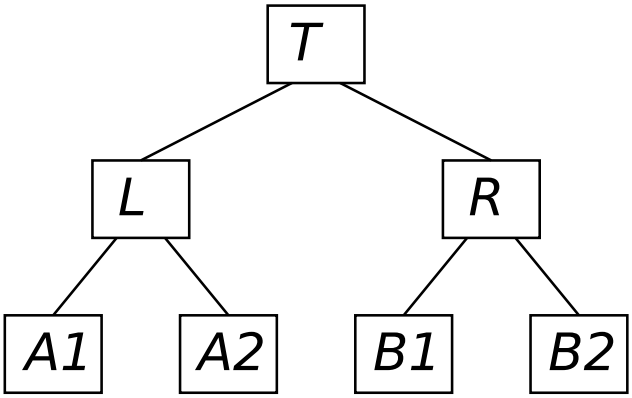
- Redo-Recovery nötig, aber durch Checkpoint begrenzt.
- Undo-Recovery nötig, aber durch Checkpoint begrenzt.
- Keinerlei Redo-Recovery nötig.
- Keinerlei Undo-Recovery nötig.
- Redo-Recovery nicht durch Checkpoint begrenzt.
- Undo-Recovery nicht durch Checkpoint begrenzt.

c) Was gilt bei Transaction-Consistent Checkpoints (multiple choice)

- Redo-Recovery nötig, aber durch Checkpoint begrenzt.
- Undo-Recovery nötig, aber durch Checkpoint begrenzt.
- Keinerlei Redo-Recovery nötig.
- Keinerlei Undo-Recovery nötig.
- Redo-Recovery nicht durch Checkpoint begrenzt.
- Undo-Recovery nicht durch Checkpoint begrenzt.

Aufgabe 9: Sperren

Geben sie an in welcher Reihenfolge Sperren angefordert werden, und, sollte eine Sperre nicht moegliche sein, kennzeichnen sie diese. Sie muessen nach einer nicht moeglichen Sperre nicht weitermachen. Teilaufgaben sind unabhaengig voneinander. Die Organisation der Daten sieht so aus:



a)

Existierende Sperren:

Datenobjekt	Sperre
R	IS
B1	S
T	IS

Transaktion fordert X-Sperre fuer B1 an:

b)

Existierende Sperren anderer Transaktionen:

Datenobjekt	Sperre
L	SIX
T	IX

Transaktion fordert X-Sperre fuer A1 an:

c)

Existierende Sperren anderer Transaktionen:

Datenobjekt	Sperre
T	IS
T	IX
B2	S
L	IS
R	IS
A2	S

Transaktion fordert S-Sperre fuer A2 an:

Aufgabe 10: Inner Join

Relation R

x	y
0	F
4	G
14	A
8	F
2	B
1	C

Relation S

x	y
14	C
3	U
6	Z
9	D
11	A
14	B
29	W

Mache einen Hash-Verbund Inner Join mit S.y = R.y, die Zwischenergebnisse der einzelnen Schritte (also das Ergebnis **nach** einem Schritt) muessen angegeben werden. Hashfunktion ist $h(k) = k$.

---

1. Lesen

Hashtabelle:

0	1

Ergebnisrelation:

S.x	S.y	R.x	R.y

2. Probing

Hashtabelle:

0	1

Ergebnisrelation:

S.x	S.y	R.x	R.y

3. Lesen

Hashtabelle:

0	1

Ergebnisrelation:

S.x	S.y	R.x	R.y

4. Probing

Hashtabelle:

0	1

Ergebnisrelation:

S.x	S.y	R.x	R.y