

## Gruppe A

Bitte tragen Sie **sofort** und **leserlich** Namen, Studienkennzahl und Matrikelnummer ein und legen Sie Ihren Studentenausweis bereit.

PRÜFUNG AUS DATENBANKSYSTEME VO 181.146			19. 06. 2006
Kennnr.	Matrikelnr.	Familienname	Vorname

Arbeitszeit: 120 Minuten. Aufgaben sind auf den Angabebältern zu lösen; Zusatzblätter werden nicht gewertet.

### Aufgabe 1:

(20)

Eine Datenbank über Musik-Aufführungen enthalte folgende Relationen:

Werk(wNr, Titel, Kategorie, Komponist) (kurz *w*),  
Orchester(oNr, Name, Stadt) (kurz *o*) und  
Auffuehrung(wNr, oNr, Datum, Ort) (kurz *a*).

Nehmen Sie an, dass  $|w| = 4000$ ,  $|o| = 500$  und  $|a| = 100000$ . Es ist die Anfrage

```
select w.Titel, o.Name, a.Datum
from Werk w, Orchester o, Auffuehrung a
where o.oNr = a.oNr
and w.wNr = a.wNr
and w.Kategorie = 'Oper'
and o.Stadt = 'Wien';
```

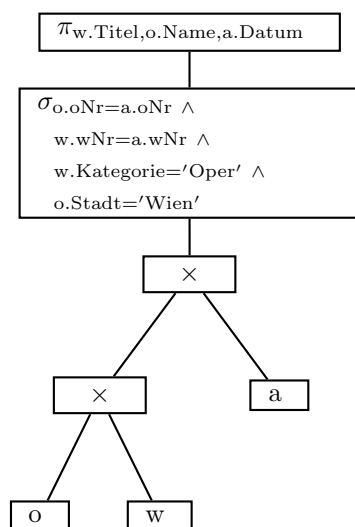
auszuführen (d.h. gesucht sind Informationen über Opern-Aufführungen von Orchestern aus Wien). Es sind die Selektivitäten  $Sel_{w/a} = 1/4000 = 0.00025$ ,  $Sel_{o/a} = 1/500 = 0.002$ ,  $Sel_{w.Kategorie='Oper'} = 0.2$  und  $Sel_{o.Stadt='Wien'} = 0.1$  anzunehmen.

(a) Zeichnen Sie ins erste Kästchen den Operator-Baum für die kanonische Übersetzung.

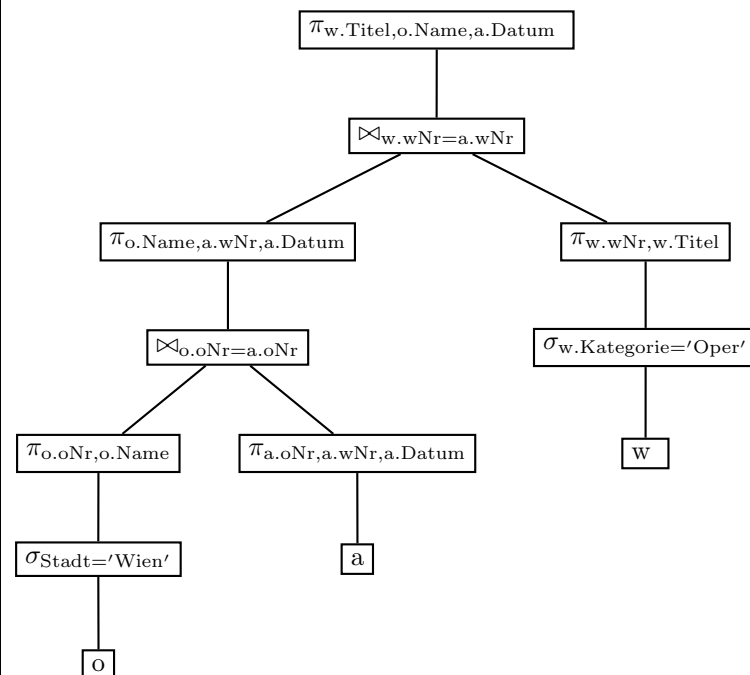
(b) Zeichnen Sie ins zweite Kästchen den Operator-Baum für den optimierten algebraischen Ausdruck. Wenden Sie für die Optimierung folgende Heuristiken an:

- Selektionen so weit wie möglich nach unten verschieben,
- Attribute, die nicht mehr benötigt werden, möglichst früh wegprojizieren,
- Kreuzprodukte durch Joins ersetzen,
- Join-Reihenfolge so, dass der erste Join ein möglichst kleines Zwischenergebnis (d.h.: möglichst wenige Tupel) liefert.

#### (a) Kanonische Übersetzung:



#### (b) Optimierter Ausdruck:



## Aufgabe 2:

(24)

Gegeben ist die folgende Historie von Transaktionen:

Schritt	$T_1$	$T_2$	$T_3$	Log: [LSN, TA, PageID, Redo, Undo, PrevLSN] or ⟨LSN, TA, PageID, Redo, PrevLSN, UndoNextLSN⟩
1	BOT			[#1, $T_1$ , BOT, 0]
2			BOT	[#2, $T_3$ , BOT, 0]
3		BOT		[#3, $T_2$ , BOT, 0]
4	$r(A, a_1)$			
5		$r(A, a_2)$		
6	$w(A, a_1 - 100)$			[#4, $T_1$ , $P_A$ , A-=100, A+=100, #1] ....
7		$r(B, b_2)$		
8			$r(A, a_3)$	
9			$w(A, 3 * a_3)$	[#5, $T_3$ , $P_A$ , A+=200, A-=200, #2] ....
10		$w(A, a_2 + b_2)$		[#6, $T_2$ , $P_A$ , A+=50, A-=50, #3] .....
11			$w(C, 2 * a_3)$	[#7, $T_3$ , $P_C$ , C+=100, C-=100, #5] ....
12		$w(B, 2 * a_2)$		[#8, $T_2$ , $P_B$ , B+=250, B-=250, #6] ....
13		$w(C, a_2 - b_2)$		[#9, $T_2$ , $P_C$ , C-=150, C+=150, #8] ....
14	commit			[#10, $T_1$ , commit, #4] .....
15			commit	[#11, $T_2$ , commit, #7] .....
16		rollback ...		[#12, $T_2$ , rollback, #9] .....
17				⟨#13, $T_2$ , $P_C$ , C+=150, #12, #8⟩ .....
18				⟨#14, $T_2$ , $P_B$ , B-=250, #13, #6⟩ .....
19				⟨#15, $T_2$ , $P_A$ , A-=50, #14, #3⟩ .....
20				⟨#16, $T_2$ , (BOT), #15, 0⟩ .....

(a) Nehmen Sie an, dass in Zeile 16 auch die Transaktion  $T_2$  mittels commit beendet wird. Ist die resultierende Historie serialisierbar?

☐ ja    ☒ nein

Wenn ja, in Reihenfolge  $T - \dots$  vor  $T - \dots$  vor  $T - \dots$ .

Wenn nein: die Historie wird durch das Streichen von zumindest 1 ..... Operationen serialisierbar.

(b) Führen Sie nun – unabhängig davon, ob die Historie serialisierbar ist – in Zeile 16 ein **rollback** für  $T_2$  durch.

Zu Beginn ist der relevante Datenbestand in der Datenbank  $A = 200$ ,  $B = 150$ , und  $C = 100$ . Tragen Sie nun das Recovery-Log zu dieser Historie (mit rollback von  $T_2$  in Zeile 16) in die rechte Spalte der obigen Tabelle ein. Dabei sind Undo/Redo-Einträge *relativ* zum Datenbestand anzugeben. Geben Sie in den Zeilen 16 ff. die Log-Einträge für die Recovery an.

Die Werte von  $A$ ,  $B$  und  $C$  nach dem rollback von  $T_2$  sind  $A : 300$  .....;  $B : 150$  .....;  $C : 200$  ......

### Aufgabe 3:

(24)

Kreuzen Sie an, ob die folgenden Aussagen wahr oder falsch sind.

1. Nehmen Sie an, dass ein Index mit 1000 Schlüsseln mittels B-Baum vom Grad 10 realisiert wird. Dann gibt es mehrere gültige Möglichkeiten, diese 1000 Schlüssel im B-Baum anzuordnen. Aber in jedem Fall hat der B-Baum die Tiefe 2 (d.h.: Wurzel + 2 zusätzliche Ebenen darunter). wahr ☒ falsch ☐
2. Betrachten Sie einen Hash Index, der die Daten selbst (und nicht nur die TIDs) enthält. Bei einer Punktanfrage mit diesem Index sind im Idealfall 1.2 Seitenzugriffe notwendig. wahr ☐ falsch ☒
3. Bei der Historie  $r_1(A)$ ,  $r_2(A)$ ,  $w_2(B)$ ,  $r_1(B)$ ,  $a_1$  führt der Abbruch der Transaktion  $T_1$  zu einem kaskadierenden Rücksetzen von  $T_2$ . wahr ☐ falsch ☒
4. Betrachten Sie folgende Historie, bei der nur BOT (= Transaktionsbeginn),  $c_i$  (= Commit) und Sperranforderungen (aber keine Datenzugriffe oder Informationen über Warten bzw. Wecken einer Transaktion) gegeben sind:  $BOT_1$ ,  $BOT_2$ ,  $lockX_1(A)$ ,  $lockX_2(B)$ ,  $lockS_1(B)$ ,  $c_2$ ,  $c_1$ . Diese Historie ist beim Zeitstempel-Verfahren mit wound-wait Strategie möglich. wahr ☐ falsch ☒
5. Betrachten Sie drei Relationen  $R(AB)$ ,  $S(AB)$  und  $T(BC)$ . Dann gilt auf jeden Fall folgende Gleichheit:  $\pi_B((R \cup S) \bowtie T) = [\pi_B(R) \cup \pi_B(S)] \bowtie \pi_B(T)$ . wahr ☒ falsch ☐
6. Wenn in einem Auswertungsplan alle Nested Loop Joins durch Hash Joins ersetzt werden, dann kann dies zu einer Verringerung der Anzahl der Tupel in den Zwischenergebnissen führen. wahr ☐ falsch ☒
7. Betrachten Sie die Kostenformel  $2 * b_R * \log_{m-1}(1 + 1)$  mit  $I = \lceil b_R/m \rceil$  für das externe Sortieren. Dabei steht  $b_R$  für die Anzahl der Tupel der Relation  $R$  und  $m$  für die Anzahl der verfügbaren Puffer-Seiten. wahr ☐ falsch ☒
8. Betrachten Sie eine Datenbank mit der Tabelle Flug (FlugNr, Fluglinie, Start, Ziel) über direkte Flugverbindungen. Alle von Wien aus per Flugzeug mit beliebig oft Umsteigen erreichbaren Orte lassen sich mittels SQL99 aber nicht mittels SQL92 abfragen. wahr ☒ falsch ☐
9. Betrachten Sie eine Datenbank mit der Tabelle Spieler (SpielerId, Name, Land, Alter, Einsaetze) aller Fußballer, die bei der WM dabei sind. Diese Tabelle sei mit Hilfe der 32 möglichen Werte des Attributs "Land" in Fragmente zerlegt. Die resultierende Fragmentierung ist horizontal, vollständig und redundanzfrei. wahr ☒ falsch ☐
10. Geschachtelte Relationen in objektrelationalen Datenbanken führen zu einer Redundanz, die zwecks Optimierung bestimmter DB-Anfragen jedoch in Kauf genommen wird. wahr ☐ falsch ☒
11. Betrachten Sie die Datenbasis-Hierarchie in Aufgabe 5. Nehmen Sie an, dass  $T_1$  die Sperren laut Aufgabe 5(a) erfolgreich angefordert hat, d.h.:  $(T_1, D, IX)$ ,  $(T_1, a_2, IX)$ ,  $(T_1, p_3, IX)$ ,  $(T_1, s_6, X)$ . Dann sind folgende Sperranforderungen (in genau dieser Reihenfolge) erlaubt:  $(T_1, p_4, IS)$ ,  $(T_1, s_7, S)$ . wahr ☒ falsch ☐
12. Betrachten Sie die Datenbasis-Hierarchie in Aufgabe 5. Nehmen Sie wiederum an, dass  $T_1$  die Sperren laut Aufgabe 5(a) erfolgreich angefordert hat. Dann ist folgende Freigabe von Sperren (in genau dieser Reihenfolge) erlaubt:  $unlock(T_1, D, IX)$ ,  $unlock(T_1, a_2, IX)$ ,  $unlock(T_1, p_3, IX)$ ,  $unlock(T_1, s_6, X)$ . wahr ☐ falsch ☒

(Pro korrekter Antwort 2 Punkte, **pro inkorrektter Antwort -2 Punkte**, insgesamt mindestens 0 Punkte)

### Aufgabe 4:

(20)

Ein Hotel-Buchungssystem verwendet eine Datenbank mit folgenden Relationen:

Hotel(HotelNr, Name, Sterne, Betten, Preis) (kurz  $h$ ),

Kunde(KundenNr, Name, Beruf, Wohnort) (kurz  $k$ ) und

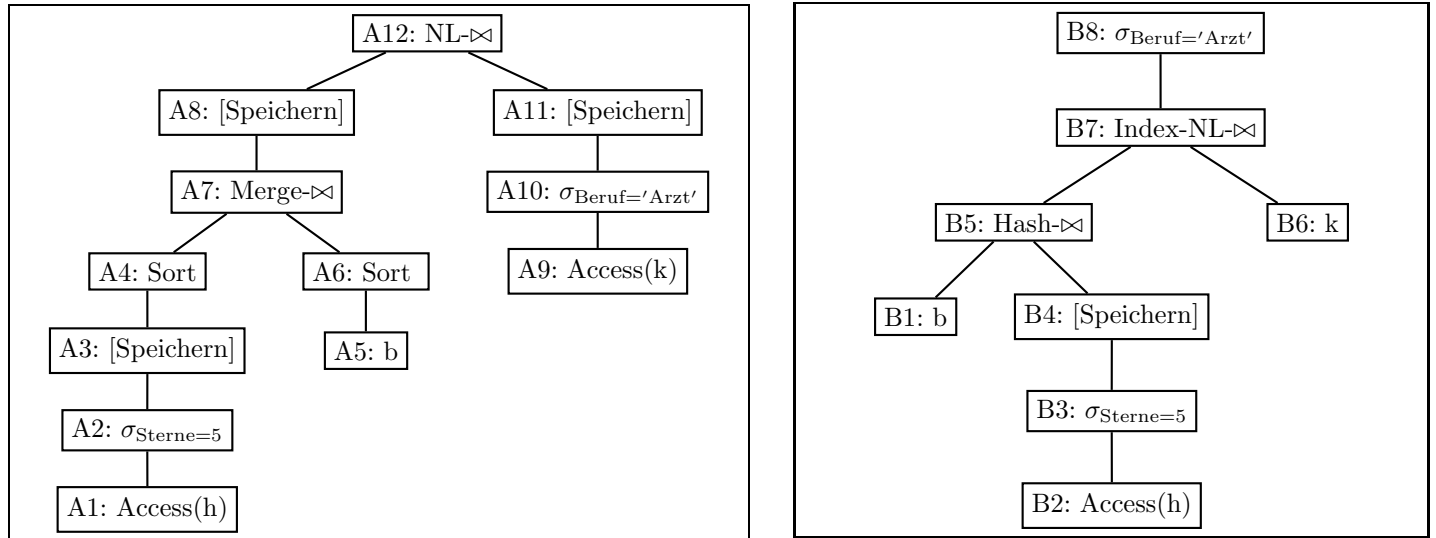
Buchung(BuchungNr, KundenNr, HotelNr, AnreiseDatum, AbreiseDatum) (kurz  $b$ ).

Nehmen Sie an, dass  $|h| = 5000$ ,  $|k| = 200000$ , und  $|b| = 1000000$ . Für die durchschnittlichen Tupelgrößen von  $h$ ,  $k$  und  $b$  sind die Werte 120, 100 und 40 Bytes anzunehmen. Nehmen Sie weiters an, dass pro Seite 2000 Bytes an Nutzinformation gespeichert werden können und dass die Hauptspeicher-Puffergröße 32 Seiten beträgt. Es ist die Anfrage

```
select *
from Hotel h, Kunde k, Buchung b
where b.HotelNr = h.HotelNr
and b.KundenNr = k.KundenNr
and k.Beruf = 'Arzt'
and h.Sterne = 5;
```

auszuführen (d.h. gesucht sind Informationen über Ärzte, die einen Urlaub in einem 5-Sterne Hotel buchen). Es sind die Selektivitäten  $Sel_{h/b} = 1/5000 = 0.0002$ ,  $Sel_{k/b} = 1/200000 = 0.000005$ ,  $Sel_{h.Sterne} = 0.2$  und  $Sel_{k.Beruf='Arzt'} = 0.1$  anzunehmen. Für die Primärschlüssel der Relationen  $h$  und  $k$  sei jeweils ein Hash-Index vorhanden. Nehmen Sie an, dass das Auslesen eines einzelnen Tupels mit einem Hash-Index durchschnittliche Kosten von 1.4 Page I/O erfordert.

Für diese Anfrage sind die Operator-Bäume für 2 Auswertungspläne gegeben: Plan A im linken Kästchen und Plan B im rechten Kästchen. Nehmen Sie bei den Block Nested Loop Joins an, dass die äußere Relation jeweils links im Baum steht – unabhängig davon, ob dies optimal ist oder nicht. Bei der Berechnung der benötigten Seiten zum Speichern einer Relation dürfen Sie vereinfachend annehmen, dass die Tupel nicht unbedingt vollständig auf einer Seite Platz haben müssen.



(a) Berechnen Sie für jeden Knoten im Operatorbaum des Auswertungsplans A eine Abschätzung für die Anzahl der Tupel im Resultat, die Tupelgröße, die Anzahl der Seiten im Resultat, und die Kosten (Page I/O). Für das Sortieren und für Joinoperationen ist auch noch die passende Kostenformel anzugeben. Tragen Sie Ihre Berechnungen in folgende Tabelle ein.

Knoten# $i$	Anzahl Tupel $T_i$	Tupel- größe $g_i$	Anzahl Seiten $b_i$	Kostenformel	Kosten (Page I/O)
A1	5000	120	300 .....	-	300 .....
A2	1000 .....	120 .....	60 .....	-	0 .....
A3	1000 .....	120 .....	60 .....	-	60 .....
A4	1000 .....	120 .....	60 .....	$2 * b_3 * (1 + I)$ mit ..... $I = \log_{31}(\lceil b_3/32 \rceil) = 1$ .....	240 .....
A5	1000000	40	20000 .....	-	0 .....
A6	1000000 ..	40 .....	20000 .....	$2 * b_5 * (1 + I)$ mit ..... $I = \log_{31}(\lceil b_5/32 \rceil) = 2$ .....	120000 ....
A7	200000 ....	160 .....	16000 .....	$b_4 + b_6$ .....	20060 .....
A8	200000 ....	160 .....	16000 .....	-	16000 .....
A9	200000	100	10000 .....	-	10000 .....
A10	20000 .....	100 .....	1000 .....	-	0 .....
A11	20000 .....	100 .....	1000 .....	-	1000 .....
A12	20000 .....	260 .....	2600 .....	$b_8 + 1 + \lceil b_8/30 \rceil * (b_{11} - 1)$ .....	549467 ....

Kosten insgesamt (Page I/O):

$$300 + 60 + 240 + 120000 + 20060 + 16000 + 10000 + 1000 + 549467 = 717127 \dots\dots$$

(b) Berechnen Sie für jeden Knoten im Operatorbaum des Auswertungsplans B eine Abschätzung für die Anzahl der Tupel im Resultat, die Tupelgröße, die Anzahl der Seiten im Resultat, und die Kosten (Page I/O). Für die Joinoperationen ist auch noch die passende Kostenformel anzugeben. Tragen Sie Ihre Berechnungen in folgende Tabelle ein.

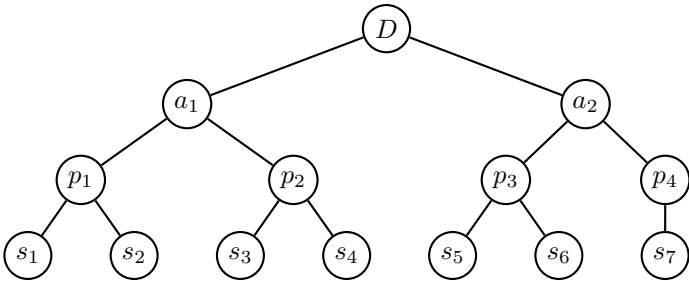
Knoten# $i$	Anzahl Tupel $T_i$	Tupel- größe $g_i$	Anzahl Seiten $b_i$	Kostenformel	Kosten (Page I/O)
B1	1000000	40	20000 .....	-	0 .....
B2	5000	120	300 .....	-	300 .....
B3	1000 .....	120 .....	60 .....	-	0 .....
B4	1000 .....	120 .....	60 .....	-	60 .....
B5	200000 ....	160 .....	16000 ....	$3 * (b_1 + b_4)$ .....	60180 ....
B6	200000	100	10000 ....	-	0 .....
B7	200000 ....	260 .....	26000 ....	$b_5 + 1.4 * T_5$ .....	296000 ....
B8	20000 .....	260 .....	2600 .....	-	0 .....

Kosten insgesamt (Page I/O):	
300 + 60 + 60180 + 296000 = 356540 .....	

**Aufgabe 5:**

(12)

Multi-Granularity Locking. Betrachten Sie folgende Datenbasis-Hierarchie.



Beantworten Sie, welche der folgenden Sequenzen von Anforderungen von Schlössern (bei zwei Transaktionen  $T_1$  und  $T_2$ ) zu Blockierungen bzw. Deadlocks führen. (Hier bedeutet  $(T_i, x, L)$ , daß Transaktion  $T_i$  versucht, Knoten  $x$  in der Hierarchie mit einem Schloß vom Typ  $L$  zu belegen.)

- (a)  $(T_1, D, IX), (T_2, D, IX), (T_2, a_1, IX), (T_2, p_1, X), (T_1, a_2, IX), (T_2, a_2, IS), (T_1, p_3, IX), (T_2, p_3, S), (T_1, s_6, X)$ :  
 Blockierung:      ja ☒    nein ☐    Deadlock:      ja ☐    nein ☒
- (b)  $(T_1, D, IX), (T_2, D, IS), (T_1, a_1, IX), (T_2, a_2, IS), (T_1, p_2, IX), (T_2, p_3, IS), (T_1, s_3, X), (T_2, s_5, S), (T_2, D, IX)$ :  
 Blockierung:      ja ☐    nein ☒    Deadlock:      ja ☐    nein ☒
- (c)  $(T_1, D, IX), (T_2, D, IX), (T_1, a_1, IX), (T_2, a_2, IX), (T_1, p_1, X), (T_2, p_3, X), (T_1, a_2, IX), (T_2, a_1, IS), (T_2, p_1, S)$ :  
 Blockierung:      ja ☒    nein ☐    Deadlock:      ja ☐    nein ☒

Gesamtpunkte: 100