Übung 4

Task 1.1.

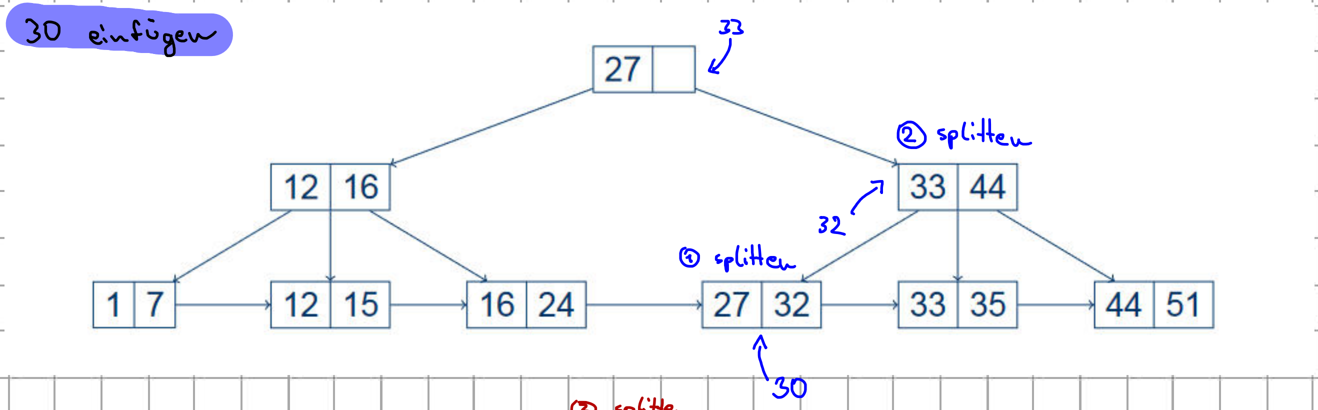
VL. 12 Folien 41 ff

durch Gleichgewicht ist Höhe vom Baum nicht mehr als $\lceil \log_{\lceil \frac{n}{2} \rceil} (k) \rceil$

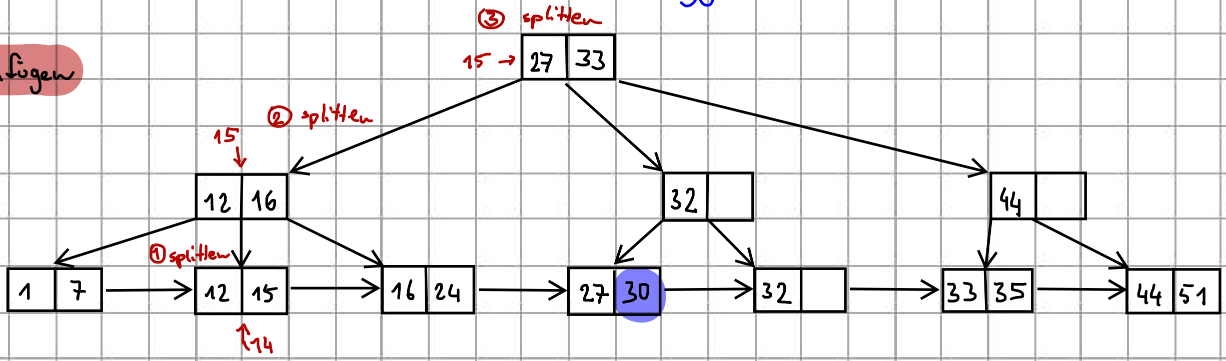
→ effiziente Suche

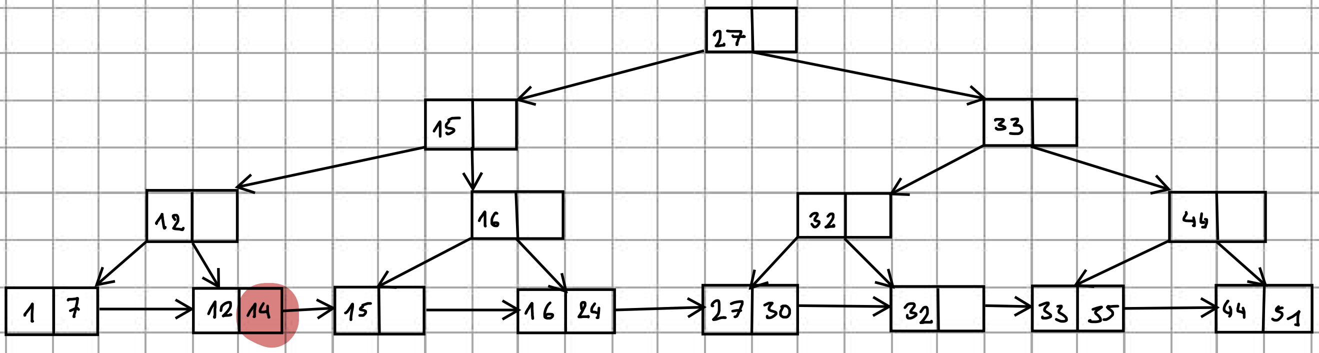
Task 1.2.

30 einfügen



14 einfügen





$S_1 = \langle r_4(x), w_2(y), w_3(z), w_1(z), r_1(x), w_4(x), w_3(x), r_1(x), w_2(z), w_4(z), w_3(y) \rangle$

$S_2 = \langle w_1(a), w_2(b), w_3(c), w_4(d), r_4(d), r_2(c), r_3(a), r_4(c), w_2(d), r_2(a), r_2(d) \rangle$

1 EN: Explain what the *ACID* properties mean in the context of transactions.
DE: Erklären Sie was die *ACID* Eigenschaften im Kontext von Transaktionen bedeuten.

2 EN: Determine all conflict pairs of S_1 and S_2.
DE: Bestimmen Sie jeweils alle Konfliktpaare von S_1 und S_2.

3 EN: Draw the precedence graph of S_1 and S_2.
DE: Zeichnen Sie jeweils den Konfliktgraphen von S_1 und S_2.

4 EN: Is S_2 serializable? If yes, specify a possible serialization of S_2.
DE: Ist S_2 serialisierbar? Falls ja, geben Sie eine mögliche Serialisierung von S_2 an.

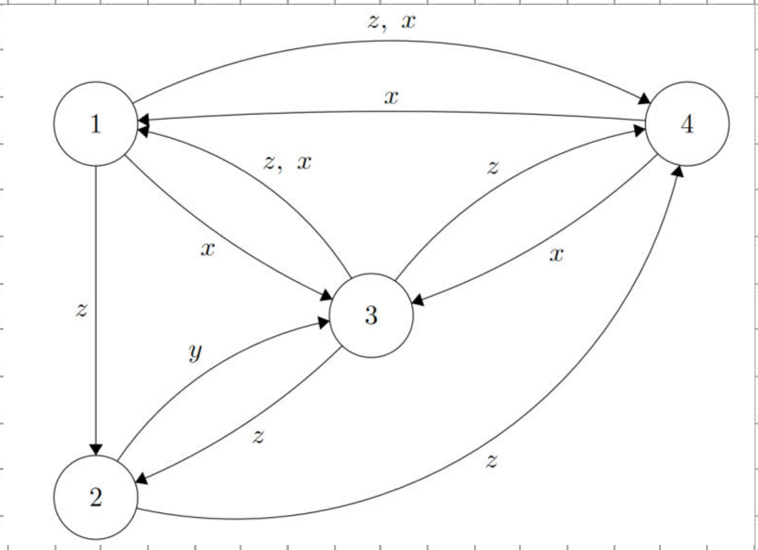
1. Die ACID Eigenschaften behalten die Integrität der Datenbank bei

- A: Atomicity (Atomarität). Entweder alle Operationen der Transaktion werden in die Datenbank übernommen oder keine (alles oder nichts Prinzip)
- C: Consistency (Konsistenz). Transaktionen hinterlassen einen konsistenten Datenbasiszustand. Dieser Endzustand muss die im Schema definierten Konsistenzbedingungen erfüllen, sonst wird die Transaktion zurück gesetzt
- I: Isolation. Transaktionen beeinflussen sich nicht gegenseitig, indem jede Transaktion so ausgeführt wird, als wäre sie die einzige. Parallele Transaktionen sind für einander nicht sichtbar
- D: Durability (Haltbarkeit). Veränderungen nach einer erfolgreichen Transaktion bleiben dauerhaft in der Datenbasis, also auch nach einem Systemfehler

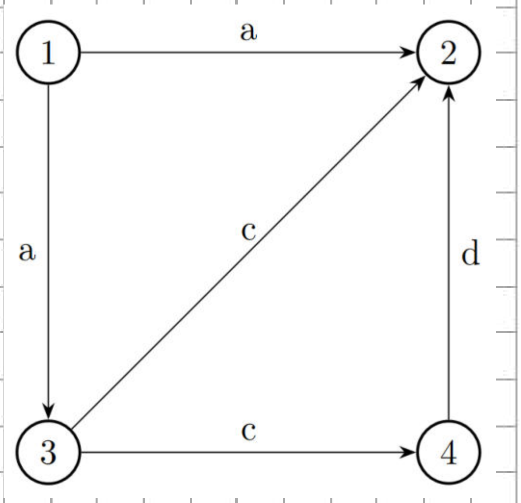
2

S1:	**S2:**
$r_4(x), w_3(x)$	$w_1(a), r_3(a)$
$w_2(y), w_3(y)$	$w_1(a), r_2(a)$
$w_3(z), w_1(z)$	$w_3(c), r_2(c)$
$w_3(z), w_2(z)$	$w_3(c), r_4(c)$
$w_3(z), w_4(z)$	$w_4(d), w_2(d)$
$w_1(z), w_2(z)$	$w_4(d), r_2(d)$
$w_1(z), w_4(z)$	$r_4(d), w_2(d)$
$r_1(x), w_4(x)$	
$r_1(x), w_3(x)$	
$w_4(x), w_3(x)$	
$w_4(x), r_1(x)$	
$w_3(x), r_1(x)$	
$w_2(z), w_4(z)$	

3 S_1

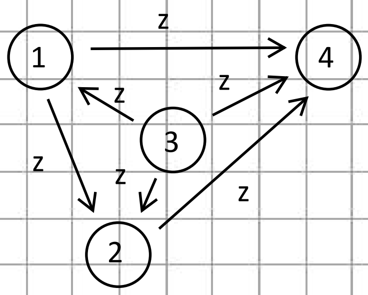


S_2



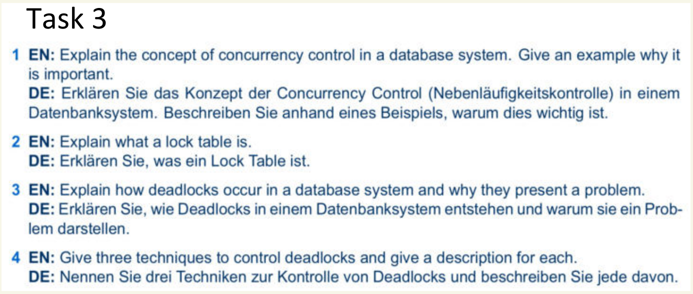
4 Nein, da Kreis in Konfliktgraph (z.B. zwischen Transaktionen 1 und 3)

wenn nur z: betrachtet wird:



Ja, ist serialisierbar, da Graph azyklisch
mögliche Serialisierung: T3→T1→T2→T4

5 Ja, ist serialisierbar, da Graph azyklisch
mögliche Serialisierung: T1→T3→T4→T2



Task 3

1 **EN:** Explain the concept of concurrency control in a database system. Give an example why it is important.
DE: Erklären Sie das Konzept der Concurrency Control (Nebenläufigkeitskontrolle) in einem Datenbanksystem. Beschreiben Sie anhand eines Beispiels, warum dies wichtig ist.

2 **EN:** Explain what a lock table is.
DE: Erklären Sie, was ein Lock Table ist.

3 **EN:** Explain how deadlocks occur in a database system and why they present a problem.
DE: Erklären Sie, wie Deadlocks in einem Datenbanksystem entstehen und warum sie ein Problem darstellen.

4 **EN:** Give three techniques to control deadlocks and give a description for each.
DE: Nennen Sie drei Techniken zur Kontrolle von Deadlocks und beschreiben Sie jede davon.

1. • ensures non-interference or isolation of concurrently executing transactions
 • e.g. multiple transactions on one bankaccount → ensures data integraty
 • "ensures that database transactions are performed concurrently without violating the data integrity of the respective databases"
 • "general area of concurrency control provides rules, methods, design methodologies and theories to maintain consistency of components operating concurrently while interacting and thus the consistency and correctness of the whole system"

2. lock table:
 • describes which locks are being hold right now
 Records: [data-item, LOCK, T] and a queue for waiting transactions

3. how deadlocks occur:
 • 2 (or more) transactions wait for each other to unlock data
 (s. Task 4)
 why problem
 • transactions will not finish

4. • Deadlock avoidance: a transaction T requesting a new lock is aborted if there is a possibility of deadlock (roll back, T rescheduled)
 • Deadlock detection: DBMS periodically tests DB for deadlocks (a victim is aborted (roll back + restart) other one continues)

- Deadlock prevention: transaction must obtain all the locks it needs before it can be executed (e.g. conservative 2PL)

Task 4

1 **EN:** Can deadlocks occur with strict 2-phase locking? If yes, how, and give an example using the table below. If not, why not?
DE: Können Deadlocks bei strengem (strict) 2-Phase-Locking auftreten? Wenn ja, wie, und gebe ein Beispiel unter Verwendung der unten stehenden Tabelle. Wenn nicht, warum nicht?

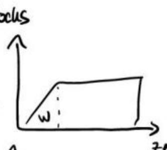
2 **EN:** Can deadlocks occur with conservative 2-phase locking? If yes, how, and give an example using the table below. If not, why not?
DE: Können Deadlocks bei konservativem 2-Phase-Locking auftreten? Wenn ja, wie, und gebe ein Beispiel unter Verwendung der unten stehenden Tabelle. Wenn nicht, warum nicht?

t	T_A	T_B
1	read-lock (x)	
2		read-lock (y)
3	write-lock (y)	
4		write-lock (x)

- A hält X und wartet auf y
- B hält Y und wartet auf X

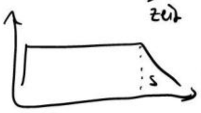
→ Deadlock

yes



alle Sperren werden erst zum EOT freigegeben

no



benötigte Sperren werden zu Beginn der TA alle auf einmal gesetzt

w = growing phase

s = shrinking phase