Prüfung: Datenbanken 1665
Prüfer: Prof. Schlageter
Dauer: ca. 20min
Note: 1,3
Termin: 25.08.2011

Die Prüfung fing etwas später an da mein Vorgänger Verspätung hatte. Zum Warmwerden sprachen wir etwas über mein Studium, dann ging es auch schnell los. Die Fragen sind fast alle in früheren Protokollen zu finden, deshalb skizziere ich die Prüfung und beschreibe nur die "neuen" Fragen.

- Es ging los mit Datenunabhängigkeit,
- was ist genau unabhängig auf der konzeptuellen Ebene, was auf der physischen.
- physische Datenorganisation
- Synchronisation, Logs, 2 Phasen Sperrprotokoll,
- Beim 2Phasen Sperrprotokoll malte er zwei verbundene Rechtecke für Konten a und b, und wollte wissen, zu welchem Zeitpunkt man genau a wieder freigeben kann.
Da kam ich nicht gleich drauf, sobald b gesperrt wurde kann man a wieder frei geben, also ziemlich am Anfang des b-Rechtecks.
- Dann wozu After-Image
- warum speichert man nicht gleich in der DB? - Zugriff mehrerer Transaktion auf eine Seite...
- wann ist das ganz besonders kritisch? - hot spots - add - sub

das wars dann ziemlich.

Fazit:
Ich kann die Prüfung empfehlen. Prof. Schlageter nimmt einem die Nervosität und ist berechenbar. Man sollte sich Synchronisation genau anschauen und sich Beispiele und Konsequenzen überlegen - einfach runterbeten reicht nicht. EAs machen, Protokolle lesen, Beispiele überlegen, dann hat man nichts zu befürchten. Viel Spass. Und Protokolle schreiben!!!

Prüfungsprotokoll Datenbanksysteme, 1665
Prüfer: Prof. Dr . Schlageter
Version: SS2008
Datum: 22.09.2008
Dauer: 15 Min.
Note: 1,0

Die Prüfungsatmosphäre war wirklich sehr locker.
Es war der typische Ablauf:

- Datenunabhängigkeit
- Serialisierbarkeit
- Wie kann man synchronisieren
- Ist lock unlock lock unlock sinnvoll?
- Was kann passieren?
- Strikte 2PL
- Was kann man auf der physischen Ebene zur Beschleunigung von Joins tun? Sortieren und parallel durchlaufen, Invertierung und Tupel, auf die die Zeiger zeigen, durchlaufen.
- Was kann man zur physischen Ebene noch sagen? Primär-, Sekundärorganisation
- Mögliche Datenstrukturen zur Speicherung?
- Beim Sekundärindex wollte er alles genau wissen, vor allem welche Nachteile sie außer dem Verwaltungsaufwand noch haben – bei z.B. 50% Selektivität. Bei 50% müssen tatsächlich fast so viele Seiten eingelagert werden, wie bei 100%.

In der Regel genügten ihm Schlagworte, aber dann hat er doch auch mal tief gebohrt. Rate zu einer eigenen Zusammenfassung und paralleles Lernen der Prüfungsberichte. Auch sehr gut fand ich die EAs 3,4,5,6 zur Kontrolle, ob man es wirklich verstanden hat. Wenn das der Fall ist, hat man glaub' nichts zu befürchten. Er sagte es wäre die einzige 1 am ganzen Vormittag gewesen (so viel zur Einschätzung der Noten anhand der alten Prüfungsprotokolle).

Kurs: 1665 Datenbanksysteme, Version WS 2004/2005
Termin: 11.2006
Dauer: < 15 Min.
Prüfer: Prof. Dr. Schlageter
Note: 1,0

Allgemein: absolut empfehlenswerte Prüfung, Prof. Schlageter ist sehr freundlich und humorvoll.

Fragen:

- Datenunabhängigkeit (logische und physische)
- Was ist in der konzeptuellen Ebene unabhängig und was nicht?
- Was ändert man, wenn man den Wertebereich eines Attributs ändert?
- Merkt man Änderungen in der physischen Ebene?
- Wie baut man eine DB bzgl. der physischen Ebene auf?
- Primärorganisation?
- Sekundärorganisation?
- Wie sieht ein Index aus?
- Wann lohnt sich ein Index, wann nicht?
- Wieso lohnt er sich bei einer Selektivität von 50% nicht?
- Serialisierbarkeit (am Beispiel Bankentransaktion)
- Inkonsistente Sicht, wie zu vermeiden?
- 2-Phasen-Sperrprotokoll
- Deadlock
- Was macht man beim Abbruch einer Transaktion?
- Thema After-Images: wieso schreibt man nicht gleich in die DB?

Viel Erfolg allen Mitstreitern!

Prüfungsprotokoll

Kurs: 1665 Datenbanken – Diplom II
Termin: 05.2005
Dauer: ca. 20 Min
Prüfer: Prof. Dr. Schlageter
Note 1.0

Allgemein:
Ich kann mich den vorliegenden Protokollen nur anschließen. Sehr angenehme ruhige Prüfung. Prof. Schlageter bringt einen auf den richtigen Weg, falls man die Frage nicht richtig verstanden hat oder man ein wenig hängt und wirkt zu jedem Zeitpunkt ruhig und freundlich.
Mitunter ist man ein wenig verwirrt aufgrund eines Themensprungs, aber dass muss wohl so sein.

➔ Empfehlenswerte Prüfung!

Mein Tip: Alles flüssig beantworten können, für jede mögliche Frage am besten auch ein Beispiel parat haben. Man sollte vermitteln können, dass man die Gesamtmaterie verstanden hat und daß einem die Zusammenhänge zwischen den einzelnen KEs klar sind.

Fragen:

- Datenunabhängigkeit
- Was ist denn in der konzeptuellen Ebene wirklich unabhängig und was nicht?
    - Unabhängig: Strukturelle Änderungen an den Relationen (Hinzufügen von Attributen, ...)
    - Abhängig: Änderung von Wertebereichen, Änderung von Beziehungstypen, ...
    - Hier sollte man dringend erwähnen, dass man die abhängigen Änderungen gegenüber den Anwendungsprogrammen bekannt machen muss.
- Wie stellt man Änderungen in der physischen Ebene im Anwendungsprogramm fest?
    - evtl. Geschwindigkeit, sonst nicht
- Wie baut man eine DB bzgl. der physischen Ebene auf?
    - Primärorganisation, Sekundärorganisation
- Was bedeutet Sekundärorganisation?
    - Indexe bzw. Invertierte Organisation
- Was steht in so einem Index? Was gibt man an den Index weiter und was bekommt man heraus?
    - Achtung: Man bekommt nicht die Ergebnissätze sondern nur Zeiger auf diese! Sollte man unbedingt korrekt ausführen.
- Wann lohnt sich solch ein Index nicht?
    - 10%- Regel
- Wieso nicht?
- Was ist eine inkonsistente Sicht?
- Können diese bei Anwendung des 2-Phasen-Modells auftreten?
- Wieso nicht?
    - Anhand des Bankenmodells (2 Konten) kann man erklären, dass entweder ein Konto bereits durch T1 gesperrt wäre (T1 liest Konto A und setzt Sperre) und T2 warten müsste, um dieses zu lesen. Andererseits kann T2 auch zuerst B lesen, was zu einem Deadlock führen würde. Beide Fälle würden nicht in eine inkonsistente Sicht führen.
- Was ist ein Deadlock? Wie stellt man dieses fest? Wie löst man diesen auf?
- Was macht eigentlich ein Optimizer?
    - Da musste ich erst mal einen Moment nachdenken, was man jetzt hören wollte. Dann fiel es mir aber zum Glück ein und erklärte die Algebraische Optimierung.
    - Graph, Optimierungsverfahren, Warum die einzelnen Verfahren gut sind? (z.B. Frühe Selektion reduziert die weiter zu verarbeitendende Anzahl an Datensätze).

... Ende und viel Glück allen Mitstreitern!

(Wer Protokolle liest, sollte auch Protokolle schreiben – gerade an einer FernUniversität)

März 05 Datenbanksysteme Professor Schlageter Note 1,3

Eine lockere Prüfungsatmosphäre kann ich nur bestätigen. Professor Schlageter ist ausgesprochen freundlich und sehr hilfsbereit, wenn man partout nicht weiterweiß (mir wollte z. B. einfach kein Beispiel zu inkonsistenter Sicht einfallen).
Natürlich muss man sich gut vorbereiten, aber wenn man aus Nervosität nicht weiterweiß, ist das kein Drama.

- Was verstehen Sie unter Datenunabhängigkeit?
  physische
  - Wenn sich beim physischen Schema etwas ändert, was bemerkt der Anwender?
    Das System arbeitet schneller oder langsamer.

  logisch
  - Was bemerkt der Anwender, wenn sich beim logischen Modell etwas ändert?
    Wertebereich, Kardinalitäten der Beziehungen (die Anwendungsprogramme müssen sich ändern, wenn die Beziehung sich von 1:1 in eine 1:n oder n:m Beziehung ändert)
- Welche Schritte werden bei dem Speichern der Daten vollzogen?
  Festlegung der Primärorganisation, der Sekundärorganisation
- Wofür sind zusätzliche Zugriffspfade wichtig?
  Man durchsucht die Datei nicht nur nach dem Primärindex.
- Wofür noch?
  Selektion
- Was ist serialisierbar?
  Wenn es eine serielle Reihenfolge der Transaktionen gibt, die das gleiche Ergebnis hat auf der Datenbank wie bei den Ausgabedaten.
- Wenn eine Transaktion Datenobjekte sperrt und wieder freigibt, sperrt und wieder freigibt, ist dies eine gute Realisierung der Synchronisation?
  Nein, es kann zu nicht serialisierbaren Reihenfolgen kommen?
- Was passiert dann?
  Inkonsistente Sicht
- Ein Beispiel?
  Banktransaktion. Buchung von Konto A auf Konto B. Dürfte man nach Buchung von Konto A beide Kontostände sehen, hätte man eine inkonsistente Sicht auf die Datenbank.
- Wie kann man das verhindern?
  Sperren
- Was kann bei Sperren passieren?
  Deadlock
- Wie erkennt man einen Deadlock und wie hebt man ihn auf?
  Wartegraph, Abbruch einer Transaktion
- Geht es auch einfacher?
  Timeout, aber dabei kann es passieren, dass kein Deadlock vorliegt.
- Nach welchen Kriterien entscheidet man sich für eine abzubrechende Transaktion?
  CPU-Zeit-Verbrauch, Kosten für Rollback, Zahl der schon angeforderten und im Speicher befindlichen Seiten
- Wenn man nicht die günstigste Transaktion nimmt, woran liegt das?
  Daran, dass diese schon oft abgebrochen wurde.
- Was passiert bei einem Abruch?
  Das Before-Image wird in die Datenbank geschrieben.
- Wozu braucht man After-Images?
  Speicherfehler und Systemfehler
- Warum schreibt man nicht gleich in die Datenbank?
  Weil immer ganze Seiten geschrieben werden.
- Warum genau?
  Weil auch andere Transaktionen auf diese Seite zugreifen.
- Wo ist das besonders schlimm?
  Bei Hot Spots

So, dann beenden wir jetzt die Prüfung.

Ich habe sehr von den alten Protokollen profitiert. Euch geht es sicher genauso, daher:
Wer Prüfungsprotokolle verwendet, sollte selbst welche schreiben!
(Auch wenn es schon viele gibt)

Prüfungsprotokoll

Kurs: Datenbanken 1665
Prüfer: Prof. Schlageter
Termin: März 2004
Prüfungsdauer: ca. 15 Minuten Aki

- „Was versteht man unter Datenunabhängigkeit?“
  (Antwort siehe Kurstext: logische und physische Datenunabhängigkeit)
- „Die Anwendungsprogramme sind also isoliert von der physischen Datenorganisation. Können sie sich vorstellen, wann der Anwender dennoch Änderungen auf der physischen Ebene bemerkt.“
  (richtige Antwort: Änderungen in der physischen Datenorganisation beeinflussen die Geschwindigkeit von Abfragen. Das bemerkt der Anwender unter Umständen)
- „Was könnten das für Änderungen sein?“
  (richtige Antwort: Unterstützungen des Verbundes auf der physischen Ebene durch Indexstrukturen)
- „Wann lohnt sich denn ein Index?“
  (1. richtige Antwort: Wenn die Selektivität hoch ist. Also weniger als 10% der Tupel zu einem Attributwert selektiert werden. (siehe Kurstext))
- „Man könnte doch sagen, dass 50% Selektivität besser ist als gar kein Index. Oder fällt Ihnen da ein Gegenargument ein?“
  (Richtige Antwort: Die Wahrscheinlichkeit, dass die Daten über den gesamten Speicherbereich verteilt sind ist groß, so dass doch fast alle Seiten in den Speicher geladen werden müssen und gegenüber der sequentiellen Suche wenig gewonnen wird)
- „Fällt Ihnen noch ein Grund ein, wann ein Index nicht lohnend ist?“
  (2. richtige Antwort: Wenn so häufig Änderungen auf den Objekten durchgeführt werden, dass sich der Verwaltungsaufwand für das Führen des Index nicht lohnt.)
- „Welches sind die Grundoperationen der Relationenalgebra?“
  (Antwort siehe Kurstext)
- „Sie kennen die Struktur einer SQL Abfrage. Wie hängt denn diese mit Grundoperationen zusammen?“ (richtige Antwort: Select = Projektion/Where = kartesisches Produkt/Where = Selektion)
- Was ist Serialisierbarkeit?“ (Antwort siehe Kurstext)
- „Welche Synchronisationsverfahren gibt es?“
  (Antwort siehe Kurstext: grundlegende Unterschiede von optimistischen und präventiven Verfahren erläutern)
- „Warum kann man dem Programmierer nicht das Management von Transaktionen überlassen?“
  (richtige Antwort: Eingriff des Programmierers kann zu Inkonsistenzen führen.)
- „Was sind denn Inkonsistenzen. Können Sie ein Beispiel dafür geben?“
  (richtige Antwort siehe Beispiel Bankentransaktion, bei der eine Transaktion T2 Daten (a und b) von Transaktion T1 liest, obwohl diese (T1) erst die aktuellen Werte für a jedoch nicht für b in die Datenbank geschrieben hat. T2 erhält eine inkonsistente Sicht und daraus entsteht dann eine inkonsistente Datenbank, wenn T2 mit dem aktuellen Wert für a jedoch mit dem veralteten Wert für b weiterarbeitet)
- „Was benötigt man bei einem Systemfehler?“
  (richtige Antwort: Um Undo und Redo zu ermöglichen benötigt man ein Log. Dabei Redo und Undo kurz erläutern.)
- Wann kommt man ohne Undo aus?
  (Antwort siehe Kurstext)

Fazit: Herr Prof. Schlageter ist als Prüfer sehr zu empfehlen. Durch Kopfnicken macht er deutlich, dass man die Frage richtig beantwortet. Man sollte das Kopfnicken nicht als Anlass nehmen, mit der Beantwortung der Frage aufzuhören, weil man meint, alles sei nun klar. Alles was man sagt, sollte aber im Zusammenhang mit der Frage stehen. Flüssiges Vortragen (z.B. Themenbereich der KE1 sowie Serialisierbarkeit, Synchronisationsverfahren, Recovery) wirkt sich positiv aus. Die Einsendeaufgaben muss man nicht bearbeitet haben, um die Prüfung meistern zu können. Da man den Kurs relativ schnell versteht, kann man leicht vergessen, dass das Verstehen für die Prüfung nicht ausreicht. Man muss es auch wiedergeben können (unter Stress). Wirklich intensiv durcharbeiten sollte man die Kapitel Synchronisation und Recovery, denn es gibt viele Interdependenzen und die stehen nicht so direkt im Kurs (hier sollte man sich auch mal die Übungsaufgaben und Lösungen am Kursende anschauen). Bei diesen beiden Kapiteln könnte auch mal ein Blick in ein anderes Lehrbuch hilfreich sein um das Ganze besser zu verstehen (z.B Elmasri, Navathe).

Gedächtnisprotokoll Kurs 1665
Datenbankensysteme

Prüfer	**:**	Prof. Dr. Schlageter
Datum	**:**	28.11.2003
Dauer	**:**	ca. 25 min
Note	**:**	2.0

- ## Datenbankenentwurf

Beim Datenbankenentwurf spielt die Datenunabhängigkeit eine wesentliche Rolle – können Sie mir sagen warum ?

Welche Abhängigkeiten kennen Sie denn ?

Ist die physische Abhängigkeit vollständig zu erreichen?
Ja ***, sonst mache das Ganze auch keinen Sinn***
Ist die logische Abhängigkeit vollständig zu erreichen?
Nein
Warum nicht, und welche Änderungen würden sich denn auch noch auswirken?
Wertebereichsänderungen eines Entity
Warum, was ändern wir denn damit?
‚Den betrachteten Teil der Welt'

- ## Physicher Datenbankentwurf

Wie gehen wir weiter vor, wenn wir nun die logischen Entities abbilden wollen.
Wir bilden diese in interne Sätze ab….
Sie denken schon zu kompliziert….wusste nicht ganz auf was er raus will ***…also meinetwegen hätten wir die internen Sätze und speichern diese gemäß einer bestimmten primären Organisation ab. Welche Möglichkeiten haben wir dann?***

Was versteht man unter Sekundärorganisation?

Wieso macht man das?

Invertierung … was gebe ich denn vor und was erhalte ich zurück?
→ Werte einer bestimmten Attributmenge (= Index)
← Tupel die in dieser Attributmenge die vorgegebenen Werte enthalten
Wir erhalten nicht die Tupel zurück sondern …?
Den Verweis auf die Tupel

- ## Synchronisation

‚Zeichnet den Kurvenverlauf f(Zeit) -> Anzahl gesperrter Objekte'…..Oft wird behauptet, dass mit danach gesperrten Objekten Serialisierbarkeit erreicht werden kann. Der Programmierer kennt aber doch seine Daten, wenn man ihm das selbst überlässt was kann dann passieren?
Lost update, inkonsistente Sicht, Phantom
Phantom kann auch das 2PL nicht beseitigen, nennen Sie ein praktisches Beispiel für eine inkonsistente Sicht
Beispiel aus Skript …. Mit R(a,b) bevor ‚b' verändert wird
Was genau ist denn nun inkonsitent
Lesende Transaktion erhält Sicht (a geändert, b noch nicht) die so gar nicht existiert

Wie sieht ihr Beispiel aus, wenn ‚man mit 2PL arbeitet. Wie werden Sperren gesetzt ?

Wenn nun die ‚lesende Transaktion' genau umgekehrt vorgeht und zuerst b liest?
…….dann kommt es durch den S-Lock der 2. Transaktion zu einem Deadlock
Wie löst man den auf, ?
Welche Transaktion wird dann abgebrochen?
Was heißt billige Transaktion?
Wann nehmen wir denn trotzdem die teuere Transaktion und brechen diese ab?
Wenn es zu einem zyklischen Neustart kommt.
Was wissen Sie über hotspots, was ist das?
Wie werden diese denn zurückgesetzt?
Warum ist ein UNDO mit den Before Images denn nicht möglich, was macht man deswegen

- **Recovery**

Was wird beim Systemzusammenbruch notwendig?

Was bedeutet denn UNDO/REDO?

Wann werden diese After Images geschrieben?

Was passiert denn mit den After Images die zwischen dem Abbruch und dem Checkpoint geschrieben werden, woher wissen wir welche wir davon einspielen müssen?
Er wollte darauf hinaus, dass man das gar nicht wissen muss, sondern einfach die After Images aller beendeten in diesem Zeitraum einspielen kann
Warum schreiben wir die After Images eigentlich nicht einfach immer direkt, vor dem Commit?
Was wird denn geschrieben auf der Datenbank?
Er wollte darauf hinaus, dass man ja ganze Blöcke schreibt und somit immer große Mengen geschrieben werden müssten

Eindruck

Auch ich kann dem allgemeinen Tenor der Gedächtnisprotokolle zum größten Teil zustimmen. Prof. Schlageter versucht mit dem Prüfling die Antworten zu ‚erarbeiten' und gibt Hilfestellungen falls Notwendigkeit besteht. Mein Problem war (wie auch schon bei Datenbanken) hin und wieder die Art der Fragestellung. Durch sie will Prof. Schlageter wohl erkennen inwieweit man Zusammenhänge verstanden hat – will aber dann auch in der Antwort genau die Richtung erkennen die er glaubt vorzugeben. Das kann vor allem dann etwas verwirren, wenn man nicht direkt erkennt welche Richtung dies sein soll. Mein Tipp wäre, sich neben dem offensichtlichen Lernstoff selbst viele Fragen zu überlegen.(Nach der Art:'… wenn Transaktion 1 lesen will was muss dann mit den Sperren passieren….wann wird was in das Logfile geschrieben und warum….usw'.
Alles in allem eine angenehme Prüfung und ein sehr netter Prüfer. Der Kurs selbst macht auch richtig Spaß und ist trotz einiger Theorie sehr praxisorientiert!

Gedächtnisprotokoll Praktische Informatik Kurs 1665
Prof. Dr. Schlageter
22.02.2002 (mit gut 30 Min Verspätung)
Dauer: ca. 15 Minuten
Note: 1,0

- Wie wird das konzeptuelle Modell in das interne Modell umgesetzt? (sequentielle, index-sequentielle Organisation, Hashing, B-Bäume, Primär- und Sekundärzugriffspfade)
- Wie kann man sekundäre Zugriffspfade realisieren? (Invertierte Organisation)
- Wann lohnt es sich das nur?(10 %-Geschichte)
- In welchen Fällen kann es sich nicht lohnen, auch wenn weniger als 10% der Tupel selektiert werden? (Felder werden so häufig geändert, daß ständige Änderungen des Index zu aufwendig wären)
- Was macht man bei einem Speicherfehler?
- Was macht man bei einem Systemfehler?
- Wie kann man ohne Before-Images auskommen?
- Wie kann man Undo vermeiden?
- Sperren auf Hot Spots erklären
- Was passiert, wenn die Euro-Umstellung auf die Hot Spots wirkt? (Multiplikation, d.h. normale Sperren statt Add- und Sub-Lock)
- Wie kann man mit Add- oder Sub-Lock abgewickelte Transaktionen dennoch zurücksetzen?
- Wie kann man einen Join physikalisch unterstützen? (invertieren und sortieren)
- Wie betreibt man algebraische Abfrageoptimierung?
- Welche unterschiedliche Wirkung haben 2PL und strikte Zweiphasigkeit und was kann passieren, wenn die Programmierer die Sperren selber setzen können? (fortgeplanzter Rollback)

Fazit: Ich kann mich den Aussagen aus den anderen Protokollen nur anschließen. Die Fragen waren nicht vorbereitet und entwickelten sich wohl auch spontan aus dem Verlauf der Prüfung. Es ging ziemliche flott kreuz und quer durch die verschiedenen Gebiete (deshalb fehlt vielleicht das ein oder andere Detail in der Liste), die überwiegend aus den anderen Protokollen bekannt sind. Es wurden aber auch neue Gebiete gestreift und einige Transferfragen gestellt, deren Antworten so nicht direkt im Kurs stehen. Kurze Denkpausen bzw. lautes Denken vor der eigentlichen Antwort werden aber akzeptiert und richtige Antworten entsprechend quittiert.

Prüfung: Hauptdiplom Datenbanksysteme 1665
Prüfer: Prof. Dr. Schlageter
Termin: 25.04.2001
Note: 1.0

Die Prüfung war sehr kurz, weniger als 15 min. Da ich anscheinend die Fragen bis dahin zur Zufriedenheit beantwortet hatte, beendete Prof. Schlageter die Prüfung sehr bald. Die Fragen waren alle aus den bisherigen Prüfungsprotokollen bekannt. Wichtig sind weniger die Details, sondern dass man die Zusammenhänge begriffen hat. Die Prüfungsatmosphäre war sehr locker; Prof. Schlageter ist unbedingt zu empfehlen.

Da die meisten Fragen in früheren Protokollen bereits beantwortet wurden, erspare ich mir die Details.

Fragen:

Wie baut man das ER-Modell zu Relationen um?
- Ich habe erläutert, wie die Entities und die verschiedenen Beziehungstypen umgesetzt werden.

Wie sieht der Join in SQL aus?
-Select ... From ...Where

Was macht man nach einem Systemabsturz?
- Neustart, Before- und After-Images erläutert

Wofür benötigz man die After-Images?
- Die Änderungen sind eventuell bei COMMIT noch nicht in der Datenbanjk

Wo sind sie dann?
- Noch im Cache.

Was passiert bei einem CHECKPOINT?
- aktive Transaktionen werden aufgezeichnet, Before-Images der abgebrochenen und After-Images der beendeten Transaktionen in die Datenbank zurückgeschrieben

Wofür sorgt das 2-Phasen-Sperrprotokoll?
- Für Serialisierbarkeit

Was kann dann nicht passieren?
- hier hatte ich etwas Probleme, worauf die Frage abzielte; Prof Schlageter verlangte dann ein Beispiel für nicht-serialisierbare Transaktionen:
R1(a), R2(b), W1(b), W2(a)
Sind nicht serialsisierbar, weil sie gegenseitig voneinander abhängig sind; T1 sperrt a, T2 sperrt b; T1 wartet auf Freigabe von b, T2 wartet auf Freigabe von a; es entsteht also ein Deadlock
Gleichzeitig streiften wir bei dieser Frage dann auch die unterschiedlichen Synchronisatuionsprobleme wie LOST UPDATE oder INKONSISTENTE SICHT

- Wie löst man einem Deadlock auf?

Protokoll Diplomprüfung 01665 Datenbanksysteme
Datum: 13.3.2000
Prüfer: Prof. Schlageter

An die genaue Reihenfolge der Fragen kann ich mich nicht mehr so genau erinnern, deshalb gliedere ich das Protokoll nach der Einteilung des Kurstextes. Ich möchte keinen Verwirren, deshalb merke ich an, dass die gegebenen Antworten nicht unbedingt richtig sein müssen.

1. Einführung und 2. Architektur eines Datenbanksystems: keine Fragen
3. Relationale Datenbanken:
    - Wie werden Beziehungstypen aus ER-Diagrammen umgesetzt?
      Abhängig vom Modell, welches man benutzen will. Für relationales Modell die Umsetzung von 1:1, 1:n, n:m Beziehungstypen erläutert.
    - Wie können diese Beziehungstypen ausgewertet werden?
      Über Abfragesprachen
    - Wie sieht die Abfrage eines Beziehungstyps aus?
      Selektion über dem kartesischen Produkt oder Join
    - Was benötigt man, wenn man einen Join einer Relation mit sich selber ausführen will?
      Tupelvariablen
    - Beschreiben Sie die zweite und dritte Normalform!
      Die Definitionen konnte ich nicht und bei der Erklärung hatte ich auch Probleme
4. Netzwerkorientierte Datenbanken: keine Fragen
5. Hierarchische Datenbanken: keine Fragen
6. Synchronisation:
    - Wofür braucht man das zwei-Phasen-Sperrprotokoll?
      Es garantiert Serialisierbarkeit von Transaktionen
    - Was garantiert das Sperren bis zum Ende der Transaktion?
      Das dass Rücksetzen von Transaktionen keinen fortgepflanzten Rollback zur Folge hat
    - Was passiert, wenn man ohne Sperren arbeitet? Geben Sie ein Beispiel was dann passieren kann!
      Beispiel für Phantomproblem und lost update aus dem Kurs genannt
    - Was sind Hot Spots?
      Intensiv genutzte Bereiche der Datenbank
    - Wie werden diese Teile gesperrt?
      Durch kurze Sperren. Wichtig ist, dass die Operationen atomar und kommutativ sind
    - Was muss man machen, um diese Operationen rückgangig zu machen?
      Die jeweilige inverse Operation ausführen
    - Warum reicht es nicht die zugehörigen Before-Images einzuspielen?
      In der Zwischenzeit könnten ja bereits andere Transaktionen ausgeführt worden sein
    - Was muss man machen, wenn man eine nicht atomare Operation ausführen will?
      Normale Sperren setzen
7. Recovery:
    - Was muss man nach einem Speicherfehler machen?
      Letztes Dump mit den entsprechenden After-Images einspielen.

    - Was passiert nach einem Systemfehler?
      Die beim letzten Checkpoint aktiven Transaktionen suchen. Das Log vom letzten Checkpoint bis zum Abbruch daraufhin prüfen, ob die Transaktionen beendet, oder neue gestartet wurden und entsprechend Undo oder Redo ausführen

- Warum ist der Checkpoint so wichtig?
  Damit man nicht das gesamte Log durchsuchen muss (mit Hilfestellung)
- Wie kann man das Schreiben der Before-Images vermeiden?
  Indem man nicht zulässt, dass eine Transaktion vor dem Commit Informationen in die Datenbank schreibt
- Wo werden diese Informationen solange hingeschrieben?
  In einen privaten Bereich (mit Hilfestellung)

8. Integrität, 9. Physische Datenorganisation, 10. Datenschutz, 11. Weitere Entwicklungen: keine Fragen

Eindruck: Prof. Schlageter legt nicht so viel Wert auf Details. Kurze Erläuterungen und die richtigen Schlagworte reichen zur Beantwortung meist aus. Allerdings sollte man einige Beispiele parat haben.

Viel Erfolg

Diplomprüfung Datenbanksysteme (01665)

Datum: März 2000
Prüfer: Prof. Schlageter
Kursversion: WS 99/00
Prüfungsverlauf:

- Abbildung ER-Modell in relationale Schemata (Insbesondere m-n Beziehung)
- 2PL mit Beispiel (Wann serialisierbar?)
- Deadlocks (Wann gibt es welche, wann nicht? Wie vermeiden?)
- Plattencrash (Was tun?)
- Systemfehler (Was tun?)
- Before Images (Wann braucht man sie nicht ?)
- Checkpoints (Warum? etc.)

Kommentar:

Die vorhandenen Protokolle geben eine gute Übersicht, auf welche Themen Prof. Schlageter Wert legt. In jedem Fall ist aber ein wirklich genaues Verständnis der Zusammenhänge in Bezug auf Checkpoints und Recovery-Mechanismen zu empfehlen. Bei diesen Dingen bringt man selbst bei guter Vorbereitung nur allzuleicht etwas durcheinander.

Prof. Schlageter hilft aber gerne weiter. Sollte man etwas zögerlich antworten, bestätigt er auch schon mal durch Kopfnicken, dass man auf dem richtigen Weg ist.

Prüfungsprotokoll Datenbanksysteme

Datum: Thu, 13 Jan 2000

Prüfung bei Herrn Schlageter in Datenbanksystemen.

Note 1,0

Prüfungsverlauf:

Wie werden Entities und Beziehungen ins Relationale Modell abgebildet?
Wie werden die Beziehungen genau modelliert (n:m, 1:n, 1:1)
Welche Basisoperationen gibt es bie der Relationenalgebra?
Als welchen Grundoperationen setzt sich ein Join zusammen?
Wie geht ein Join in SQL?
Wann ist eine Schedule serialisierbar?
Nennen Sie eine Beispiel-Schedule, die das 2PL-Protokoll verletzt und nicht serialisierbar ist
Welchen Vorteil hat Sperren bis EOT?
Wann braucht man UNDO?
Wie kann man UNDO vermeiden?
Hot-Spots, welche Sperren?
Wie kann man z.B die "Add"-Funktion rückgängig machen?
Welche Synchronisationsverahren gibt es noch ausser Sperrverfahren?
Wie funktioniert das Optimistische Verfahren?
Wie erkennt man, ob die Transaktion zurückgesetzt werden muß?
Speicherfehler (Platte kaputt). Was ist zu tun?

Prüfung begann pünktlich. Atmosphäre sehr angenehm. Kurze Unterhaltung - Nervosität vergessen. Wer nicht hingeht ist selber schuld.

Kurs	Prüfer	Datum:
1665 Datenbanken 1666 verteilte Datenbanken	Prof. Schlageter	17. November 1999; 14:30 Uhr

- Wieso hat man bei Datenbanken eine "Schichten-Architektur"?

Ich erzählte etwas für das "Ansi 3-Schichten" Modell und erkläre das externe, interne und konzeptionelles Schema. Ich sagte, daß die wichtigste Konsequenz aus der "Ansi 3-Schichten Architektur" die sei, daß Datenunabhängigkeit *(logische und physische)* dadurch erst möglich gemacht werden kann.

- Und wie sieht das dann bei verteilten Datenbanken aus?

Ich erwähnte, daß man hier aufgrund der "Vernetzung" noch ein globales Schema.

- Wieso verwendet man überhaupt verteilte Datenbanken?

Gründe genannt, wie im Kurs!

- Wieso gibt es überhaupt Transaktionen

Sind und Zweck von Transaktionen erklärt.

- Was gibt es für Synchronisationsverfahren?

Optimistische und pessimistische Verfahren genannt

- Wie stellt man im optimistischen Verfahren Serialisierbarkeit sicher?

Ich habe den Abhängigkeitsgraph erwähnt, und gesagt, daß es aber praktisch mit dem 3-Phasenmodell sichergestellt wird. Alle drei Phasen erklärt.

-Und warum ist das den nun Serialisierbar?

Irgendwie konnte ich mit dieser Frage nichts anfangen. Nach kurzer Diskussion schrieb Prof. Schlageter – nach einem kurzen Ausflug zu den Sperrverfahren – folgendes auf: "lock X, ..., unlock X, ..., lock Y, unlock Y". Er wollte ganz konkret wissen, wieso dies nicht serialisierbar ist. Die Antwort war einfach: diese Transaktion verstößt gegen das 2-PL Protokoll!

- Und wie macht man das im verteilten Fall?

Erklären des 2PC-Protokolls, sowie das man noch Synchronisationsverfahren benötigt.

- Wie optimiert man Anfragen bei vDB?

Grundgedanken formuliert. Über Partitionierung gesprochen. Lokale Ausführung auf Teilknoten, etc. Kostenaspekt erwähnt. Letztendlich "Join-Methoden" angesprochen.

Wie funktioniert nun der Semi-Join? Wieso ist der so gut?

Aufgezeigt wie der Semi-Join funktioniert. Erwähnt, daß der Kerngedanke die Reduzierung der zu übertragenden Daten an die beteiligten Knoten ist. In diesem Zusammenhang kurze Erläuterung der entsprechenden "Hash" Methode.

Wann ist der Aufwand für die Verbundberechnung mittels Semi-Join größer? Wann kann das passieren?

Ich antwortete, daß dies immer dann der Fall ist, wenn sich die bei der Relation beteiligten Tupel nur geringfügig verkleinern lassen, so daß der betriebene Aufwand nicht im Verhältnis zur Kosteneinsparung steht. Hier meinte Prof. Schlageter, daß dies eine typische "Kohl-Antwort" sei und dies nicht viel aussagt. Ebenso meinte er, daß ich überrascht sein werde, daß dies bei "besonders kleinen Relationen" der Fall ist.

Fazit

Obwohl meine Leistung deutlich unter meinen Erwartungen lag, bleibe ich bei meiner bereits im Protokoll zu "Datenstrukturen" vertretenen Meinung:

"Prof. Schlageter ist ein sehr freundlicher Prüfer. Meiner Meinung nach legt Prof. Schlageter sehr viel Wert darauf, ob man ein Gebiet verstanden hat und dieses auch kommunizieren kann. Dabei ist ihm offensichtlich bewußt, daß - zumindest einige der Fragen - bereits aus alten Prüfungsprotokollen hinreichend bekannt sind. Insofern eignen sich die Prüfungsprotokolle hervorragend zur Vorbereitung. Prof. Schlageter ist uneingeschränkt zu empfehlen."

Meine Unsicherheit und unpräzise Aussage auf die Frage zur Serialisierbarkeit ist wohl die Ursache für die schlechte Benotung. Ebenso vertrat Prof. Schlageter die Meinung, daß ich "die Dinge lediglich auswendig gelernt" habe, also gewisse Lerninhalte nicht verstanden habe. Auch "rede ich zuviel".

Wie dem auch sei! Mein Eindruck ist einfach, daß an diesem Tag die "Chemie" zwischen Prüfling und Professor nicht "gepaßt" hat und ich wohl noch ein paar Tage mehr Vorbereitung benötigt hätte.

Hauptdiplomsprüfung Datenbanksysteme 1665
Datum: 19.04.1999
Version der Vorlesung: WS 98/99
Dauer: ca. 15 min
Prüfer: Prof. Schlageter
Beisitzer: Frank Laskowski
Note: 1,3

Relationenalgebra: Welche Operationen gibt es?

Vereinigung
Differenz
Karthesisches Produkt
Projektion
Selektion

Speicherfehler
Was tun?

Neue Platte einbauen
Dump und After-Images

Braucht man alle After Images?
Nein, nur die von den wirklich bei dem Absturz sicher abgeschlossenen Transaktionen

Wir haben uns ziemlich lange bei After-Images aufgehalten. Prof. Schlageter hat da sehr gebohrt, weil ich wegen seiner Fragestellung etwas verwirrt worden bin. Auf jeden Fall war das, was er hören wollte, daß man wirklich sicher sein muß, daß die After-Images, die man durchläuft, auch von Transaktionen stammen, die ordnungsgemäß beendet wurden, um sich nicht eine inkonsistente Datenbank einzuhandeln.

Was sind Before - Images, wann werden sie geschrieben?

Bevor Änderungen von nicht abgeschlossenen Transaktionen in der Datenbank durchgeführt werden können, muß das Before-Image ins Log geschrieben werden.

Undo-Redo erklären.

In Checkpoints die zur Zeit aktiven Transaktionen eintragen.
Schreiben der Before-Images der abgebrochenen Transaktionen und After-Images der abgeschlossenen Transaktionen in die D.B. schreiben.

Gibt es auch eine Möglichkeit, ohne die Before-Images auszukommen?

Indem man No- Undo / Redo oder No - Undo / No - Redo benutzt.

Wie funktioniert das?
Man muß gewährleisten, daß eine nicht abgeschlossene Transaktion niemals in die Datenbank schreiben darf.

Gedächtnisprotokoll mdl. Diplomprüfung Kurs 1665

am 15.3.99
Prüfer: Prof. Schlageter
Dauer: 15 min
Note: 1.0

- Was macht man bei einem Speicherfehler?
  Dump einspielen, After-Images aus dem Log einspielen.
- Was macht man bei einem Systemfehler?
  (Je nach Recovery-Strategy) Log vom letzten Checkpoint ab vorwärts durchlaufen f. REDO, vom Crashzeitpunkt ab rückwärts bis z. Checkpoint f. UNDO.
- Welche Transaktionen müssen beim UNDO betrachtet werden?
  Diejenigen, die zum letzten Checkpoint n.n. abgeschlossen waren.
- Wann benötigt man keine UNDOs?
  Wenn die Transaktionen auf ihrem eigenen Speicherbereich arbeiten und erst bei COMMIT in die Datenbank schreiben.
- Welches Synchronisationsverfahren kommt ohne UNDOs aus?
  Das optimistische: Arbeitsphase, Validationsphase, Schreibphase.
- Wie wird in der Validationsphase Serialisierbarkeit geprüft?
  Entweder durch Mitführen eines Abhängigkeitsgraphs oder (einfacher!) durch Überprüfung, ob eine andere Transaktion eines der Objekte während der Arbeitsphase geschrieben hat.
- Was kann beim Zwei-Phasen Sperrprotokoll passieren?
  Fortgepflanzter Rollback, falls nicht bis EOT gesperrt wird.
- Wie sperrt man Objekte auf hot-spots?
  Durch "kleinere" Sperren als im Zwei-Phasen-Protokoll, add-lock, sub-lock für atomare Operationen add und sub.
- Was braucht man dann bei der Recovery?
  Inverse Operationen.
- Was passiert, wenn nun eine andere Operation (z.B. Multiplizieren) auf ein hot-spot-Objekt zugreifen will?
  Sie muss ein normales S- oder X-Lock gemäß dem Zwei-Phasen-Protokoll setzen, d.h. man kann die Vorteile der kleinen Sperren nicht mehr nutzen.
- Welchen Grundoperationen der Relationenalgebra entspechen die SQL-Befehle SELECT, FROM und WHERE?
  SELECT = Projektion, FROM = Kartesisches Produkt, WHERE = Selektion.
- Was benötigt man für einen Join einer Relation auf sich selbst?
  Tupelvariable
- Nennen Sie ein Beispiel, wo ein solcher Auto-Join nötig ist!
  Relation ANGEST(NAME,GEHALT,VORGESETZTER_NAME) und die Abfrage: "Welcher Angestellte bekommt mehr Gehalt als sein Vorgesetzter?".

Eigentlich ist den Kommentaren aus den bereits existierenden Protokollen fast nichts mehr hinzuzufügen: Prof. Schlageter ist ein sehr sympatischer und fairer Prüfer. Ich hatte mich bei zwei Fragen ziemlich verlaufen (bei den Checkpoints hatte ich die zum Checkpoint n.n. abgeschlossenen Transaktionen mit den zum Crashzeitpunkt n.n. abgeschlossenen verwechselt und bei den hot-spots musste mir Prof Schlageter schon sehr auf die Sprünge helfen, bis ich erklären konnte, welche Sperre eine nicht zu add kommutative Operation setzen muss). Dennoch gab dies offensichtlich keinen

Diplomprüfung Datenbanksysteme (01665)

Datum: 02.03.99

Prüfer: Prof. Dr. Schlageter

Note: 1.0

Dauer: ca. 15 min

Welche Grundoperationen hat die Relationenalgebra?
Vereinigung, Differenz, Kart. Produkt, Projektion, Selektion

Was ist ein Join?
Definition + Darstellung durch Grundoperationen
Welchen Operationen in der Relationenalgebra entspricht ein SQL-Befehl?
SELECT - Projektion
FROM - kart. Produkt
WHERE - Selektion
Wie stellt man einen Join in SQL dar?
Man nutzt Darstellung durch Grundoperationen der Relationenalgebra aus.
Was macht man bei einem Speicherfehler?
Neue Platte kaufen, Recovery: Dump einspielen, Informationen aus Logfile, soweit noch vorhanden, einspielen.
Welche Informationen aus dem Log braucht man?
After-Images der abgeschlossenen Transaktionen.

Wann werden diese geschrieben?
Wenn eine Transaktion COMMIT ausführen will.

Das kann man auch dann machen, wenn die Daten geändert werden. Wie läuft das dann mit dem Recovery?
Einspielen der After-Images abgeschlossenen Transaktionen.

Wie funktionieren optimistische Synchronisationsverfahren?
Arbeits-, Validierungs- und Schreibphase erklärt.

Was geschieht genau in der Validierungsphase?
Abhängigkeitsgraph erstellen und auf Zyklus prüfen ...

Das geht noch einfacher.
Prüfen, ob die Objekte, die die Transaktion gelesen hat, von anderen Transaktionen geändert wurden.

Kann ein Deadlock auftreten?
Nein, bei optimistischen Verfahren nicht, da ja nicht gewartet wird.

Sperrverfahren, was versteht man unter Sperrhierarchie?
Sperrhierarchien und Intensionssperren erklärt.

Was weiß man, wenn eine Relation gesperrt ist?
Man darf in einer höheren Hierarchiestufe keine Sperre mehr setzen.

Was passiert bei einem Deadlock?
Man muß ihn erkennen und auflösen. Time-Out-Verfahren und Deadlockerkennung mit Hilfe des Wartegraphen erklärt.

Wann ist eine Transaktion "billig"?
Wenn ihr Abbruch und Neustart am wenigsten kostet, d.h. am wenigsten Datenbankzugriffe und CPU-Zeit beansprucht. Kein oder wenig fortgepflanzter Rollback.

Wann muß man fortgepflanzten Rollback machen?
Wenn die abgebrochene Transaktion das ACID-Prinzip verletzt (Isolation). Andere Transaktionen konnten die Änderungen der Transaktion schon lesen und müssen daher auch abgebrochen werden.

In der Praxis wird das vermieden, wie?
Verhindern, daß andere Transaktionen Zwischenergebnisse sehen können.Transaktionen arbeiten auf privaten Speicherbereichen. Erst mit COMMIT sind die Änderungen für andere Transaktionen sichtbar.

Diplomprüfung Datenbanksysteme 1665

Version: Kurs WS 98/99

Prüfer: Prof. Schlageter

Termin: April 1999

Dauer: ca. 15 min

Note: 1,7

- Was tut man, wenn die Festplatte defekt ist? (Speicherfehler)

- Nach diesem Stichwort quer durch Recovery, Backup, Logs, Before-/Afterimage

- Sperren, Sperrprotokolle: 2Phasen, Strikte 2Phasigkeit
  Sperrmodi, S-, X-Locks, Add-, Sub-Locks, bei diesen kommutativen Sperren ist Kompensation nötig

- Normalformen: 1., 2., 3., BC-Normalform

Das Prüfungsverhalten ist nach den vorliegenden Protokollen gut einzuschätzen. Auch die Themen wiederholen sich, so daß man bei der Vorbereitung die Stoffmenge eingrenzen kann.

Bei meiner Prüfung war ich überpünktlich da, aber ein paar Minuten zu spät, sonst wäre ich direkt, noch vor dem Termin, dran gekommen.

So hatte Prof. Schlageter eine Besprechung begonnen, so daß ich warten mußte und erst mit 20 min Verspätung geprüft wurde.

Die Atmosphäre ist locker, Prof. Schlageter versucht, einem die Nervosität zu nehmen. Er ist als Prüfer uneingeschränkt zu empfehlen.

Viel Erfolg bei Eurer Prüfung!

Sperren bei Hot-Spots

Bei Hot-Spots muß man nicht „mit Kanonen auf Spatzen schießen“. Man kann, wenn man kommutative, atomare Operationen identifizieren kann, mit weniger restriktiven Sperren auskommen. Im Beispiel des Skripts war von Sub-Lack und Add-Lack die Rede.

Und was ist, wenn ich zwischendurch multiplizieren muß?

Dann muß ich doch wieder mit „normalen“ Locks arbeiten, denn das ist dann keine kommutative Operation mehr.

Wie kann man Serialisierbarkeit noch gewährleisten?

Frage zielte auf Optimistische Synchronisationsverfahren ab;

Ich habe dann erst mal erzählt, was optimistische Synchronisationsverfahren überhaupt sind und wie sie funktionieren. Dann kam ich auf die Validationsphase und auf die Serialisierbarkeit von Transaktionen.

$RS(T_i)$:= Menge der von Transaktion Ti lesenden Transaktionen
$WS(T_i))$:= Menge der auf Transaktion Ti schreibenden Transaktionen

T_i validiert erfolgreich ⇔ Für alle Transaktionen T_j, die in der Arbeitsphase von T_i geschrieben haben, gilt
$RT(T_i) \cap WS(T_j) = \emptyset$

Normalformen, wofür braucht man die?
Zur Anomalievermeidung

Welche Anomalien gibt es denn?
Beispiel für eine geben.
Einfüge-Anomalie
Lösch-Anomalie
Änderungs-Anomalie

Beispiel mit dem Abonnieren einer Zeitschrift aus dem Skript genannt.

Nennen der Normalformen

Habe die Normalformen aufgezählt, aber die 3. und BNCF in der Negativformulierung angegeben, also, es darf keine funktionale Abhängigkeit zwischen Nichtschlüssel und Nichtschlüsselattribut geben.

Kommentar:

Die Prüfung selbst findet in sehr angenehmer Atmosphäre statt. Erst einmal wurde ich nach meinem Uni-Werdegang gefragt und was ich so beruflich mache etc.
Den Eindruck eines lockeren Gesprächs, wie es so in einigen Protokollen beschrieben worden ist, kann ich nur unterstreichen.

Prof. Schlageter hat bei mir allerdings bei einigen Fragen relativ gründlich nachgebohrt. Speziell bei den After-/Before-Images hat er mich aus der Enge, in die er mich getrieben hatte, nicht so schnell entlassen.

Jedoch spricht es für seine Prüfungserfahrenheit, daß ich, obwohl der kleine Lapsus ziemlich am Anfang der Prüfung passiert ist, nicht nervös geworden bin, denn darauf hat mich der Professor erst einmal reden lassen, weil er wohl gemerkt hat, daß ich nervös zu werden drohte.

Diplomprüfung Datenbanksysteme

Kurs 1665

Datum:8.7.98

Prüfer: Prof. Dr. Schlageter

Note:1.0

Dauer:<15 min

- Was passiert (machen wir) bei einem disk-head-crash?
- Transaktionsmanager, Rekonstruktion, Dump/Backup, logisches-physikalisches Log, Before- After-Images...
- Was passiert (machen wir) bei einem Speicherfehler?
- (NO)UNDO-(NO)REDO auch bei den verschiedenen Synchronisationsverfahren
- Synchronisationsverfahren
- Serialisierbarkeit, Optimistisch, Validation!, Abhängigkeitsgraph,
- Pessimistisch, Deadlock, Deadlockauflösung, TimeOut, Rücksetzen, Preclaiming, Strikte Zweiphasigkeit Fortgesetzter Rollback!
- Join in SQL
- SELECT ATTRIBUT1 FROM TABELLE1,TABELLE2 WHERE TABELLE1.ATTRIBUTn=TABELLE2.ATTRIBUTm ....
-

Die Prüfung began etwas füher, da ich schon auf dem Flur wartete. Prof. Schlageter ist sehr nett, das Thema leicht, die Prüfung auch nicht schwer. Wer sich in Datenbanken bei Prof. Schlageter nicht prüfen lässt ist selbst schuld!

In der Virtuellen Universität gibt es im Demo-Kurs zu Datenbanksystemen ein Video des Professors (8 MB), ein Glossar und einige wenige zusätzliche Aufgaben.

Wer Gedächtnisprotokolle nutzt, sollte selbst welche schreiben!

Prüfungsprotokoll

1665 - Datenbanksysteme

Prof. Dr. Schlageter

23. April 1998

ca. 15 min

Fragen

- **Wie setzt man ein ER-Modell in eine relationale Datenbank um?**
  Auflösung von Entities und Beziehungen in Tabellen erklärt, insbesondere 1:1-, 1:n- und n:m-Beziehungen
- **Wozu dient die Normalisierung?**
  Vermeidung von Anomalien; an einem Beispiel die Probleme, die auftreten, wenn keine 2. Normalform vorliegt, erläutert; Bedingungen der 3. Normalform (keine transitive Abhängigkeit von Nicht-Schlüsselattributen), der Boyce-Codd-Normalform (keine transitive Abhängigkeit von Schlüsselattributen) und der 4. Normalform (keine zwei Multi-Wert-Abhängigkeiten) erklärt
- **Nennen Sie die Grundoperationen der Relationenalgebra!**
  Vereinigung, Differenz, kartesisches Produkt, Projektion, Selektion (Als ich auch noch Join nannte, kam die Nachfrage *"Join ist **keine** Grundoperation. Mit Hilfe welcher Grundoperationen kann ein Join ausgedrückt werden?"* kartesisches Produkt mit anschließender Selektion gemäß Join-Bedingung
- **Wie wird in SQL ein Join ausgedrückt?**
  Syntax erläutert und die verschiedenen Join-Varianten erklärt (inner, outer, left, right, full join); Einführung von Platzhaltern bei Join einer Tabelle mit sich selbst
- **Welche Maßnahmen als Vorsorge für einen Plattencrash notwendig?**
  regelmäßiger Plattendump, insbesondere bei großen Datenbanken inkrementeller Dump; Einsatz von Plattenspiegelung (Nachfrage: *"Wird dadurch ein Dump überflüssig?"* Nein, da ja unter unglücklichen Umständen beide Platten gleichzeitig fehlerhaft werden können.); Log-Files
- **Wie läuft dann ein Recovery ab?**
  der aktuellste Dump wird eingespielt, anschließend werden anhand der After-Images in der Log-Datei Redo-Operationen für alle abgeschlossenen Transaktionen durchgeführt
- **Wann muß die Redo- bzw. Undo-Information in die Log-Datei geschrieben werden?**
  Redo (After-Image) bei Commit; Undo (Before-Image) bevor eine tasächliche Änderung der Datenbank erfolgt
- **Was sind Hotspots?**
  sehr intensiv parallel genutzte Bereiche der Datenbank; Einsatz spezieller Sperren statt Zwei-Phasen-Sperrprotokoll; Operationen auf Hotspots sind kommutativ

Eindruck

Das Unangenehmste an der Prüfung war, daß sie mit einer halben Stunde Verspätung begann, das machte sich bei mir in Form gestiegener Spannung bemerkbar. Die Prüfung selbst lief aber in sehr lockerer Atmosphäre ab und erinnerte eher an ein nettes Plauder(viertel)stündchen. In den Fragestellungen wurde nur der zu erläuternde Themenkomplex angerissen. An den Stellen, wo Prof. Schlageter etwas präziser ausgeführt haben wollte, kamen entsprechende Nachfragen. Kleinere Unsicherheiten werden toleriert (bis ich z.B. den Join in Form von Grundoperationen formuliert hatte, habe ich - laut denkend - erst mal alle verkehrten Operationen kombiniert, bis

ich schließlich die richtige Antwort gegeben habe).
Sehr anzuraten ist es, sich alte Prüfungsprotokolle anzusehen; dann kann man thematisch kaum unangenehme Überraschungen erleben.
Aus meiner Sicht ist Prof. Schlageter als Prüfer uneingeschränkt zu empfehlen!
Viel Erfolg!

Mündliche Diplomprüfung - Praktische Informatik

Prüfer : Prof. Dr.Schlageter
Datum : Februar 1997
Kurs : 01665 Datenbanksysteme

?? Wenn auf einmal der Strom ausgeht, was kann da passieren und was ist zu tun ?
- ? ?*Themenbereich Recovery, und zwar hier: Systemfehler incl. Neustart*
- ? ?*Integritätsverletzung der Datenbanken*
- ? ?*hierzu UNDO / REDO nötig*
- ? ?*hierzu notwendige Recovery-Maßnahme: zusätzliche Daten auf dem Hintergrundspeicher in Form eines Log-File*
- ? ?*verschiedene Recovery-Strategien möglich*

?? Was benötigt man außer dem Log-File, wenn man Transaktionen " bearbeiten" muß ?
- ? ?*Checkpoints*

?? Was steht in den Checkpoints und was wird beim Checkpoint gemacht ?
- ? ?*Liste der aktiven Transaktionen*
- ? ?*alle BEFORE-Images abgebrochener Transaktionen werden in die Datenbank geschrieben*
- ? ?*alle AFTER-Images abgeschlossener Transaktionen werden in die Datenbank geschrieben*

?? Kann man auf REDO verzichten ?
- ? ?*ja, wenn man sicherstellt, daß eine Transaktion vor dem COMMIT in die Datenbank schreibt*
- ? ?*ob UNDO bzw. REDO ist abhängig von der gewählten Recovery-Strategie*

?

?? Welche Verfahren zur Synchronisation kennen Sie ?
- ? ?*optimistische (verifizierende, validierende) Synchronisationsverfahren*
- ? ?*pessimistische (präventive) Synchronisationsverfahren*
  - ?? *Sperrverfahren*
  - ?? *Zeitstempelverfahren*
- ? ?*Erklärung der Grundzüge der Verfahren*

?? Was kann bei einem optimistischen Synchronisationsverfahren nicht eintreten ?
- ? ?*Deadlock (kurze Erklärung)*

?? Wie sieht die Abbildung von SQL-Queries in Relationenalgebra aus ?
- ? ?*SELECT = Projektion*
- ? ?*FROM = Kartesisches Produkt*
- ? ?*WHERE = Selektion*

Allgemein: Prüfung lief in gelockerter Atmosphäre in Form eines Prüfungsgespräches ab; hier entwickelten sich auch die meisten der oben angesprochenen Punkte.

1. Prof. Schlageter, 23.03.98, ca. 15 min.

Welche Synchronisationsverfahren gibt es?

Transaktion - wie funktioniert das (Setzen v. Sperren, Freigabe ect.)?

2-Phasen- Sperrprotokoll/Strikte 2-Phasigkeit (genau).

Was ist der Vorteil bei Strikter 2-Phasigkeit?

Wann können Deadlocks auftreten? Wie erkennt man sie und wie kann man sie beheben?

Was sind die Kostenkriterien einer Transaktion?

Was ist ein JOIN? Wie wird ein JOIN in SQL abgefragt?

2. 1665, Prof Schlageter, 07.05.97, Dauer: ca. 20 min, 1,3

Sehr angenehme Prüfungsatmosphäre. Die Fragen sind nicht ,,vorgefertigt" sondern ergeben sich eher spontan.

- Entwurf im relationalen Modell:
  Abb. der Objekte im konzeptuellen Modell auf Objekte im relationalen Modell.
  Realisierung von Beziehungen (1 : n, n : m) im relationalen Modell.
  Entwurfsproblem: Erläuterung der Anomalien an einem Beispiel.
- Vergleich SQL-Klauseln mit den Operationen der Relationenalgebra:
  SELECT => Projektion
  FROM => kartesisches Produkt
  WHERE => Selektion
- Relationenalgebra: Verbundoperationen erklären (Theta erwähnen!).
- Synchronisation:
  Welche Verfahren gibt es?
  Optimistisches Verfahren näher erläutern.
  Beispiel: Während eine Transaktion aktiv ist wird ein Flugschein ausgedruckt. Danach wird die Transaktion abgebrochen. Welche Probleme entstehen dabei?
  => Flugschein in der Rand, Reservierung verloren.
- Recovery:
  Welche Log-Verfahren gibt es?
  Welche Informationen werden beim physischen Log benoetigt?
  Recovery-Strategien: speziell UNDO/NO-REDO und NO-UNDO/NO-REDO erklären.
  Speicherfehler und entsprechende Recovery-Maßnahmen erklären.

3. Prof. Schlageter, Oktober 1995, ca. 20 min, Note: 1,0

Allgemeines:

Herr Schlageter stand etwas unter Zeitdruck (45 min Verspätung), war aber dennoch sehr gut gelaunt. Er prüfte nur das Nötigste und stellte mehrere t*rickreiche* Fragen, um gleich zu Beginn zu sehen, ob der Prüfling die Kurse [11]nur[11] auswendig kann,

oder ob er den Stoff ,,richtig" kann.

- was passiert beim Commit?

(quer durch den ganzen Kurs. ..(Images, Locks, Rollback,...))

Sperrverfahren - wann ist kein UNDO oder REDO nötig?

optimistische Synchronisationsverfahren erklären - Validationstest?

physische Optimierung:
- was für Möglichkeiten?
- welche Operationen sind ,,teuer"? (kart.Produkt)

- wie kann man den Join schneller als das kart. Produkt ($O(n^2)$) machen? (beide Listen sortieren ($O(n*log(n))$), dann parallelen Durchlauf)

4. Prof. Schlageter im Mai 1996

- Wie werden die Beziehungen des ER-Diagramms in Relationalen Datenbanken realisiert?

- Was ist zu tun, wenn die Festplatte defekt ist?

- Gegeben sei folgende Situation:

```
        Checkpoint                          Abbruch wegen ,,Defekt"
T1          |                                   |
|-----------|-----------------------------------|
            |          T2                       |
            |          |--------------------------|
```

- Welches Image braucht man bei T 1, welches bei T2?
- Welche Teile von T 1 müssen nochmals ,,nachgefahren" werden und welche nicht? Warum?
- Wie sieht ein Join in SQL aus?

- Wozu braucht man Tupelvariablen?

- Was ist ein Deadlock? Welche Strategien gibt es hierbei?

- Was kann bei ungünstiger Deadlockbehandlung passieren?

- Was sind „Optimistische Verfahren“?

- Erläuterung der einzelnen Verfahren.

5. Prof. Schlageter, April 1996, ca. 15 Minuten

Was ist eine Transaktion und was geschieht wenn eine Transaktion beendet wird?

Freigabe von Sperren etc.

Welche Synchronisationsverfahren gibt es?

Welche Rolle hat das Betriebssystem?

Erläuterung Zwei-Phasen-Sperrprotokoll.

Was passiert wenn eine Transaktion ,,Abort" ausführt?

Welche Transaktionen müssen beim Rücksetzen berücksichtigt werden?

Wie wird Serialisierbarkeit geprüft?

Wie werden Entities und Beziehungen im relationalen Datenmodell dargestellt?

Wie werden m : n - Beziehungen dargestellt?

6. Prof. Dr. Schlageter, Juli 98, Dauer: <15 min.,1,0

- Was passiert (machen wir) bei einem disk-head-crash?
- Transaktionsmanager, Rekonstruktion, Dump/Backup, logisches-physikalisches Log, Before- After-Images...
- Was passiert (machen wir) bei einem Speicherfehler?
  - (NO)UNDO-(NO)REDO auch bei den verschiedenen Synchronisationsverfahren
- Synchronisationsverfahren
  - Serialisierbarkeit, Optimistisch, Validation!, Abhängigkeitsgraph,
  - Pessimistisch, Deadlock, Deadlockauflösung, TimeOut, Rücksetzen, Preclaiming, Strikte Zweiphasigkeit Fortgesetzter Rollback!
- Join in SQL
  - SELECT ATTRIBUT1 FROM TABELLE1,TABELLE2 WHERE TABELLE1.ATTRIBUTn=TABELLE2.ATTRIBUTm ....

Die Prüfung began etwas füher, da ich schon auf dem Flur wartete. Prof. Schlageter ist sehr nett, das Thema leicht, die Prüfung auch nicht schwer. Wer sich in Datenbanken bei Prof. Schlageter nicht prüfen lässt ist selbst schuld!

7. **Prof. Dr. Schlageter, Oktober 96, 15 Minuten, 1.0**

Welche Sychronisationsverfahren gibt es in DBS?
Erklären Sie das optimistische Verfahren!
Was geschieht genau während der Validationsphase?
Welches Sperrverfahren garantiert Deadlock-Sicherheit?
Was macht man, wenn bei der Anwendung von Sperrverfahren ein Deadlock auftritt?
Was verstehen Sie genau unter einer ‘billigen’ Transaktion?

8. Prüfer: Prof. Güting, Feburar 96, 35 min !, 1,0

Prüfungseindruck:
Prof. Güting ist ein guter Prüfer! Er versucht dem Prüfling die Nervosität zu nehmen, indem er ihn auffordert zu einem vorgegebenen Thema das anzubringen, was der Prüfling weiß. Es kommen kaum Zwischenfragen, die einen unterbrechen, sondern weiterführende bzw. vertiefende Fragen (siehe Prüfungsverlauf).
Die bevorzugten Themen sind:

- Relationen (Definition, R.Algebra/-Kalkül)
- Anfrageformulierung in R.Algebra/-Kalkül
- ER-Modell

- Netzwerk-DB (insbesondere N-M Relationen im Netzwerkmodell)

Ein globales Verständnis, insbesondere der Zusammenhänge, ist wesentlich wichtiger als Details der einzelnen Kurseinheiten zu kennen!

Prüfungsverlauf:

1. Nennen Sie ein gängiges Datenmodell?
   Relationenmodell
2. Was ist eine Relation?
   - allg. Beschreibung
   - Definition
3. Nennen Sie Abfragesprachen
   - R.Algebra
   - R.Kalkül
4. R.Kalkül näher erläutern. Beispiele angeben!
   Vorgabe von Prof. Güting:
   Relationen: **Städte**(Sname, Bev, Land), **Länder**(Lname, LBev, Partei)
   Abfrage: "Gib die Großstädtename (Bev > 500.000) in deren Landesregierung die Partei xxx mitregiert"
   Antwort:
   Range Länder X
   $\{Städte.SName | \exists X(X.LName = Städte.Land \wedge X.Partei = 'xxx' \wedge Städte.Bev > 500.000)\}$
5. Was sind hier frei und gebundene Variablen?
   - X ist gebunden
   - 'Städte' ist frei (wichtig, da sonst keine Ergebnissmenge)
6. Wie ist es mit der Mächtigkeit des R.Kalküls? Ist SQL gleichmächtig?
   - Ich zu SQL: 'JA'
   - Güting: 'SQL ist mächtiger! - Warum?'
   - Ich: 'Unter Betrachtung der reinen Kalküloperationen ist es nicht mächtiger - aber uner Berücksichtigung der Funktionen/Operatoren wie z.B. Sort, Group by, Sum, AVG ...ist es mächtiger'
7. Kann man in R.Algebra einen Sortieroperator definieren.
   - Nein, da RA auf Menge operiert! (nicht auf Listen)
   - Güting: "Also arbeitet SQL nicht richtig mit dem Relationenmodell, wenn man es genau nimmt. siehe auch Duplikate bzw. deren Entfernung
8. Erklärung vom Netzwerkmodell
   Beispielmodellierung zu Städte<->Länder
9. Was ist wenn mehrere Parteien in der Landesregierung sind?
   n:m Beziehung notwendig => Zwischensatz (logischer) im Netzmodell - im Gegensatz zum ERM [Städte] <--[Länder] -->[LandPartei]<--[Partei]
10. Wie erfolgt die Auswertung obiger Abfrage?
    Navigation, etc. (Detail-Beschreibung)
11. Wie sieht die Kommunikation zwischen Anwendungsprog. und Netz-DB aus?
    UWA erläutern (wichtig: Zeiger für Currency)
12. Was stört an obigen Relationen?
    'Länder' nicht in 2NF. Erläuterung der Anomalien, Normalformen allg. erläutert
13. Warum ist 'Länder' nicht in 2NF?
    LBev ist nicht voll funktional abhängig vom Schlüssel sondern nur von Lname als Schlüsselteil!

14. Was muß also getan werden?
    Zerlegung in Länder(LName, Lbev) und L-Partei(LName, Partei)

9. Fragenkatalog

KE1 - Einführung - Architektur eines Datenbanksystems

Was ist eine Datenbank und was unterscheidet sie von einem Filesystem?

Eine Datenbank ist eine integrierte Ansammlung von Daten, die allen Benutzern eines Anwendungsbereiches als gemeinsame Basis aktueller Information dient. Mit Integration ist hier gemeint, daß die Daten entsprechend den natürlichen Zusammenhängen der Realwelt strukturiert sind und daß bestimmte Aspekte der Realwelt modelliert werden. Die Nutzung erfolgt durch viele Benutzer (die jeweils eine individuelle Sicht auf die Daten erhalten). Die Kontrolle liegt komplett beim DBMS.

Bei der traditionellen Datenverwaltung mit Filesystemen definiert jeder Anwender seine eigenen Dateien (mit Sätzen die Felder enthalten). Daraus resultiert Redundanz, dadurch teures Update und Daten-Programmabhängigkeit, Inkonsistenzen sind sehr wahrscheinlich, insgesamt ist dieser Ansatz sehr inflexibel. Bei einer Datenbank hingegen erfolgt die Verwaltung der Daten zentral und es wird eine Unabhängigkeit der Daten von den Anwendungsprogrammen erreicht.

Was versteht man unter Datenunabhängigkeit?

Darunter versteht man die Unabhängigkeit von Anwendungsprogramm und Daten. Unter logischer Datenunabhängigkeit versteht man die Isolierung der Anwendung von Änderungen des konzeptuellen Modells (es ist keine Änderung der Anwendung beim Einfügen eines zusätzlichen Attributes oder eines Beziehungstyps nötig). Physische Datenunabhängigkeit hingegen bedeutet die Isolierung der Anwendung von der physischen Datenorganisation d.h. dem internen Modell (Hash, B-Tree, Heap oder z.B. Anlegen von Indizes - Dateninversion)

Welche Änderungen im konzeptuellen Schema wirken sich auf die Anwendungsprogramme aus und welche nicht: Einfügen eines Attributes, Einfügen eines Entitätstyps, Änderung des Wertebereichs eines Attributes von Integer in Real?

Nur die Wertebereichsänderung wirkt sich aus. Die übrigen Änderungen wirken sich wegen der Datenunabhängigkeit zwischen Daten und Anwendung nicht aus.

Beschreiben Sie das Schichtenmodell eines Datenbanksystems!

Sogenanntes Dreischichtenmodell:

- konzeptuelle Ebene: logische Gesamtsicht, Zusammenfassung der Daten und ihrer Beziehungen auf logischer Ebene, Integritätsbedingungen, Beschreibung durch einen Datenbeschreibungssprache (data description Language = DDL)
- interne Ebene: physische Datenorganisation; Aufbau der Sätze , Zugriffsmethoden für Sätze, Indizes, Verkettungen, interne Repräsentation der Daten - Speicherstruktur
- externe Ebene: View für die/den Benutzer, Bearbeitung der Daten durch Datenmanipulationssprache (data manipulation language = DML)

Aufgaben eines DBMS:
- Definition einer DB
- Aufnehmen, Verändern, Löschen, Selektieren (Abrufen) von Daten
- Prüfung und Sicherstellung der Integrität
- Recovery nach Systemabstürzen
- Koordination mehrerer Benutzer
- Datenschutz
- Bearbeitung von Anfragen: Transformation externes-konzeptuelles-internes Schema
- Kommunikation mit dem Betriebssystem
- Steuerung des Systempuffers (DB-Cache)
- Übergabe der Daten und von Statusinformationen (z.B. in User Working Area).

Wie heißt das Modell für die konzeptuelle Ebene?

ER-Modell: Entity (Entitätsmenge): Unterscheidbares Objekt der Realwelt, z.B. Projekt, Kunde, Mitarbeiter (Entität: lat. das Dasein eines Dinges gegenüber seiner Wesenheit)
Relationship (Beziehung): z.B. istMitarbeiterVon.
Attributierter Beziehungstyp: Attribute werden einer Beziehung zugeordnet. Jeder Entitätstyp besitzt Attribute, z.B. Mitarbeiter: name, pers_nr ....
Schlüssel: Attributmenge um eine Entität zu identifizieren. Komplexität von Beziehungstypen: 1:1 (z.B. verheiratet), 1:n (ist Chef von), n:m (Student belegt Kurs)

Wie werden Entitätsmengen und Beziehungen im ER-Diagramm dargestellt?

Entitätsmengen: Rechtecke; Beziehungen: Rauten + Kanten, die die beteiligten Entitätsmengen verbinden.

Wie werden Operationen auf Entitätsmengen im ER-Diagramm dargestellt?

Beziehungen zwischen Entities: $B(E_1,E_2,...,E_k)$, Beziehungen sind meist zweistellig

Wie erfolgt die Bearbeitung eines DML-Befehls?
- Interpretation durch das DBMS
- Ermittlung der zugehörigen konzeptuellen Beschreibung
- Transformation auf internes Schema/ Ermittlung der Zugriffspfade/Queryoptimierung
- Ermittlung der Datenseite (Page)
- prüfen ob Seite im Systempuffer(Cache) enthalten ggf. Anforderung vom Betriebssystem
- Transformation der gelesenen Daten durch das externe Schema und Bereitstellung in der UWA (user work area) des Anwendungsprogramms
- Bereitstellung von Statusinformation

KE2 - physische Datenorganisation

Wie sieht ein Index aus?

Ein Index ist eine Abbildung einer Attributkombination auf Satzzeiger. DB-Tabelle A besitzt einen Index für Attributkombination B ó Tabelle A ist invertiert bezüglich B. Indextabelle: Tabelle mit Einträgen der Form (Attributwert, Zeigerliste). Indexsequentielle Organisation: mehrstufiger Index

(=Index für Index); Einfügen: bei nicht dichtem Index Speicherreserve, ggf. Überlaufbereich ('overflow pages'), in bestimmten Abständen ist eine Reorganisation nötig

Wie ist ein B-Baum definiert?

Die Knoten eines B-Baumes sind Speicherblöcke gleicher Länge: Für einen B-Baum der Ordnung k gilt je Block mind. k, max 2k Sätze (außer der Wurzel); jeder Satz besteht aus einem Schlüssel K und einem Nichtschlüssel Teil W; Schlüssel sind aufsteigend sortiert; innerer Knoten mit m Schlüsseln hat m+1 Söhne; alle Blätter haben dieselbe Höhe h, e.g. alle Pfadlängen von der Wurzel zu den Blättern sind gleich.

Wie kann man Verbindung zwischen Sätzen unterstützen?

- durch Invertierung (Erzeugung von Indizes).
- durch Listen und Ringe
  Die zu verbindenden Sätze werden in Listenstrukturen abgelegt.
  (schwierig bei n:m Beziehungen)
- durch variable Zeigerfelder
  An jeden Satz wird eine Zeigerfeld variabler Länge angehängt, mit dessen Hilfe auf die zu verbindenden Sätze verwiesen wird.
- durch Kettsätze
  Bei n:m Beziehungen zwischen Sätzen wird ein neuer Satztyp eingeführt in dem jedes Beziehungspaar durch einen Satzeintrag repräsentiert wird

KE3 - Relationale Datenbanken I

Was ist das relationale Datenmodell?

Daten werden als Relationen (entsprechen Tabellen, Entitätsmengen) dargestellt; Zeile entspricht Entity, Spalte entspricht Attribut, Schlüssel: minimale Attributmenge die ein Tupel RÍ D_1 x D_2 x .... x D_n; D_i: Domains, identifiziert. Relationsschema R(A_1,....,A_n) spezifiziert eine Relation R mit den Attributen A_1,....,A_n; jedem Attribut A_i ist ein Wertebereich dom(A_i) zugeordnet.

Welche Abfragemöglichkeiten gibt es?

- Relationenalgebra: Definiert Operationen auf Relationen, die neue Relationen erzeugen.
  Grundoperationen: Vereinigung , Mengendifferenz, kartesisches Produkt, Selektion (Auswahl bestimmter Tupel), Projektion(Auswahl bestimmter Spalten)
  Weitere Operationen: Join, Natural Join.
  Beispiel: (ANGEST[ANGNR=ANGNR]ANG_PRO)[PNR=17][NAME]
- Relationenkalkül: Durch ein Prädikat (eine Bedingung) wird die Menge der gewünschten Tupel beschrieben;
  Beispiel: {t|q}, dabei ist t eine Liste von Attributnamen, q: Prädikat, z.B. {ANGEST.WOHNORT|ANGEST.BERUF='Programmierer'}

Welche Unterschiede gibt es zwischen Relationenalgebra und Relationenkalkül?

Die Relationenalgebra stellt konstruktive Regeln zur Erzeugung neuer Relationen bereit, hier insbesondere die Grundoperationen (Vereinigung, Differenz, kart. Produkt, Selektion, Projektion) und zusätzlich die Operationen Join und Nat. Join.

Der Relationenkalkül dagegen arbeitet mit deskriptiven Regeln. Mit seiner Hilfe wird die Menge der bei einer Abfrage gewünschten Tupel beschrieben

indem eine Bedingung angegeben wird, die die Tupel erfüllen müssen.

Sind Abfrageergebnisse im Voraus beweisbar?

Bei der Relationenalgebra ist eine sichere Bestimmung des Ergebnisses möglich, da konstruktive Regeln verwendet werden.

Formulieren Sie bitte eine Abfrage in Relationenalgebra und in Relationenkalkül!

Relationenalgebra: ANGEST[Name='Meier'][Gehalt]

Relationenkalkül: {ANGEST.Gehalt|ANGEST.Name='Meier'}

Wie werden Entitätstypen und Beziehungen in relationalen Datenbanken dargestellt?

Beide werden als Relationenschemata dargestellt.

Was ist das Ergebnis einer Selektion/Projektion auf einer Relation?

Wiederum ein Relation.

Was ist die teuerste Operation der Relationenalgebra?

Der unqualifizierte Join ist die teuerste Operation, denn er ergibt das kartesische Produkt zweier Mengen und bildet somit bei zwei Tabellen der Mächtigkeiten n und m eine neue Relation der Mächtigkeit nm.

Wie stellt man den Join in SQL dar?

Seien $R(A_1,.....,A_n)$ und $S(B_1,....,B_m)$ Relationen.

In Relationenalgebra: $R[A_i=B_j]S$

In SQL: SELECT $A_1,.....,A_n,B_1,....,B_m$ FROM R, S WHERE $A_i=B_j$

Wie stellt man den Natural Join in SQL und in Relationenalgebra dar?

Seien $S(s_1,..,s_n,a_1,..,a_m)$ und $R(r_1,..,r_k,a_1..a_m)$ Relationen mit gleichen Attributennamen $a_1..a_m$ die sowohl in S als auch in R vorkommen.

- SQL:
  Select $S.s_1,...,S.s_n,S.a_1..S.a_m,R.r_1,..,R.r_k$ Where $S.a_1=R.a_1,..,A.a_m=R.a_m$
- Relationenalgebra:
  R NATJOIN
  $S:=(RxS)[S.a_1=R.a_1,..,A.a_m=R.a_m][S.s_1,...,S.s_n,S.a_1..S.a_m,R.r_1,..,R.r_k]$

Wie stellt man einen Join zwischen einer Relation mit sich selbst dar?

Diese Auto-Join genannte Operation gelingt mit Hilfe der Verwendung von Tupelvariablen:

Relationenkalkül:

RANGE ANGEST X

RANGE ANGEST Y

{Y.Name| (X.Gehalt > Y.Gehalt) Ù (X.Name = Y.Vorgesetzter)}

Relationenalgebra:

Y[Y.Gehalt > X.Gehalt][X.Name = Y.Vorgesetzter]X [Y.Name]

SQL: SELECT Y.Name FROM Angest X, Angest Y WHERE X.Name = Y.Vorgesetzter AND Y.Gehalt > X.Gehalt

Wozu benötigt man Tupelvariablen?

Tupelvariablen sind immer dann nötig , wenn der Verbund einer Relation mit sich selbst gebildet werden soll. (siehe 2.3.10) oder dann, wenn bei einem Join von zwei verschiedenen Tabellen Attribute mit gleichen Namen auftreten. Anders ausgedrückt: Immer dann wenn innerhalb einer Abfrage verschiedene Tupel derselben Relation betrachtet werden, braucht man Tupelvariable.

Wohin wird SQL übersetzt?

SQL wird direkt vom Datenmanager interpretiert.

KE4 - Relationale Datenbanken II

Wozu dienen Normalformen?

Normalformen dienen der Vermeidung von Anomalien und zur Vermeidung von Redundanz.
Beispiel:
LIEFER(Lieferant, Artikel,Adresse, Preis)
Einfüge-Anomalie: Lieferant ohne Artikel (bisher keine Lieferung) kann nicht berücksichtigt werden.
Lösch-Anomalie: Löschen eines Artikels führt möglicherweise auch zum Verlust des Lieferanten.
Änderungsanomalie: Bei einer Preisänderung eines Artikels muß die ganze Relation durchsucht werden

Wie sind die verschiedenen Normalformen definiert und wie kann man sie anschaulich erklären?

1.NF: Die Attributwerte sind unteilbar. (d.h. keine mengenwertigen Attribute)
2.NF: Jedes Nichtschlüsselattribut A von R ist voll funktional abhängig von jedem Schlüssel S von R (und nicht nur von Teilen des Schlüssels)
3.NF: Für alle implizierten Abhängigkeiten S® A mit A Ï S gilt: S enthält einen Schlüssel oder A ist Schlüsselattribut(d.h. es gibt keine Abhängigkeiten zwischen Nichtschlüsselattributen einer Relation)
BCNF (Boyce Codd Normalform): Für jede funktionale Abhängigkeit S® A gilt: S enthält einen Schlüssel für R. (d.h. es gibt keine Abhängigkeit zwischen Nichtschlüsselattributen und Schlüsseln)

Was bedeutet funktionale Abhängigkeit?

Sei $R(A_1,...,A_n)$, X,Y Í $\{A_1,...,A_n\}$
Y ist voll funktional abhängig von X: X® Y: es gibt keine Relation vom Typ R in der 2 Tupel denselben Wert für X aber verschiedene Werte für Y haben (==wenn 2 Tupel denselben Wert für X haben, dann haben sie auch denselben Wert für Y).
Fd-Menge=Menge der funktionalen Abhängigkeiten (funcional dependencies); F^+=Closure von F=Menge der funktionalen Anhängigkeiten die von F impliziert werden.

Was muß man bei der Ausführung von SQL-Anfragen tun?

- Optimierung der Anfrage
- Transformation in konzeptuelles und internes Schema
- Beschaffen der benötigten Sätze und Transformation in externes Schema; Realisierung durch Datenmanager (Mengenschnittstelle); Zugriffsmanager (Ein-Tupel-Schnittstelle); Systempuffer(Cache)Manager (Seitenschnittstelle)

Wie werden relationale Datenbanken implementiert? Schichten eines DB-Systems?

- Datenmanager: Mengenschnittstelle, Übersetzt und optimiert die Anfragen, Integritäts- und Zugriffskontrolle
- Zugriffsmanager: 1-Tupel-Schnittstelle, Einfügen/Löschen/Suchen/ Ändern von Tupeln, Operatoren für Schemata, Transaktionsmanagement
- Systempuffermanager: Seitenschnittstelle (bei Bedarf Anforderung von neuen Seiten vom Betriebssystem; pinned pages (Seiten die im Cache sind und nicht beliebig in die DB geschrieben werden dürfen), forced output (Seiten müssen ‘durchgeschrieben’ werden))

Wie erfolgt die Adressierung für physische Sätze?

Man unterscheidet physische, seitenbezogene, logische Adressierung. Die Adresse besteht aus zwei Komponenten: Adresse der Datei und Position innerhalb der Seite. Der Zugriffsmanager ermittelt die benötigte Seite , diese wird vom Systempuffermanager angefordert, der diese bei Bedarf durch einen Betriebssystemaufruf vom Externspeicher anfordert.

Wie werden Tupeladressen berechnet?

Der Datenmanager fordert Tupel mit bestimmten Attributwerten an, die Adressen werden über Indizes vom Zugriffsmanager bestimmt.

Wie erfolgt die Optimierung von Abfragen? Was ist bei Projektionen zu beachten?

- Selektionen möglichst früh vornehmen um die Ergebnismenge klein zu halten
- Zusammenfassung von Selektionen zu komplexen Selektionen da dies die Zahl der Zugriffe auf die Daten minimal hält.
- Bei Projektionen zuerst diejenigen Attribute nehmen, die keine Eliminierung von Duplikaten erfordern, dann möglichst früh (also wenn mindestens ein Schlüssel des Relationenschemas erhalten bleibt), sonst möglichst spät.
- Suche nach gemeinsamen Teilbäumen im Operatorbaum und Vereinigung dieser Bäume (Blätter: Relationen, innere Knoten Operationen, Selektion, Join, Projektion)
- Joins möglichst spät
- Optimierung auf physischer Ebene: Ausnutzung von Sekundärindizes (Invertierung), Sortierung, Speicherstruktur der Datenbanktabellen

Wie wird ein Join ausgeführt?

Beispiel: R(A,D), S(B,C); Gesucht R[A=B]S

- einfach und ineffizient: für jedes Tupel in R mit A=a wird S durchsucht nach B=a
- mit Sekundärindex: Index weist jedem a die Tupelmenge in S mit B=a zu
- mit Sortierung: R und S werden sortiert (bezgl. A bzw. B), dann werden R und S parallel durchlaufen.

KE5 - Netzwerkorientierte/Hierarchische Datenbanken, Transaktionen

Was sind Netzwerkdatenbanken?

Logischer Satz (entspricht der Entität im Rel. Modell), Felder des Satzes (Attribute), Beziehungstypen: Set-Typ (B:Owner-Typ, Member-Typ), entspricht 1:n Beziehungstyp (beim Netzwerkmodell gibt es nur 1:n Beziehungstypen) vom Owner-Typ zum Member-Typ, Satzweise Verarbeitung; Navigieren mit den Sets; Currency-Konzept: Bezug von DML-Befehlen auf aktuelle Positionen im Netzwerk.

Wie werden bei Netzwerkdatenbanken Beziehungen und Entitäten dargestellt? (Vergleich zu relationalen Datenbanken)

Entität mit ihren Attributen entspricht dem logischen Satz mit Feldern. 1:n Beziehungstyp entspricht Set-Typ (vom Owner-Typ zum Member-Typ); n:m Beziehungstyp wird indirekt dargestellt: E-E’ => E-EK-E’ (analog zu Kettsätzen); im Unterschied zu relationalen Datenbanken wird bei Netzwerkdatenbanken im Netzwerk navigiert, d.h. mit dem Currency-Konzept auf den jeweils aktuellen Satz zugegriffen.

Wie lautet die Definition der Serialisierbarkeit von Schedules?

Ein paralleles System von Transaktionen ist serialisierbar genau dann, wenn

es eine serielle Ausführung derselben Transaktion gibt , die denselben Datenbankzustand und dieselben Ausgabedaten wie die Summe der Einzel-Transaktionen erzeugen.

Was bedeutet ACID-Prinzip?

Atomicity, Consistency, Integrity, Durability von Transaktionen

Was passiert alles bei einem Commit?

- After-Images müssen zu diesem Zeitpunkt im Log stehen
- Before-Images können verworfen werden
- Datenbank kann muß aber nicht geschrieben sein (Daten können sich auch noch im Systempuffer befinden)
- Daten werden für die Außenwelt sichtbar

Was sind die Kostenkriterien einer Transaktion?

CPU-Intensität, I/O-Intensität

Kann eine Transaktion zwischen BOT und EOT ein Dokument mit rechtlichem Charakter erzeugen?

Nein, die Daten sind erst nach EOT bzw. nach dem abschließenden Commit 'amtlich'. Falls eine Dokument zwischen BOT und EOT amtlich wäre, dann wäre im übrigen die Rücksetzbedingung (siehe 2.6.5) verletzt.

Welche Sychronisationsverfahren für Schedules gibt es?

- verifizierend (optimistisch, validierend): Vor dem Commit wird geprüft, ob eine Transaktion an einer nichtserialisierbaren Schedule beteiligt ist; falls ja, dann wird die Transaktion abgebrochen. (abort)
  Die Prüfung auf Serialisierbarkeit wird auch certification bzw. validation genannt.
- präventiv (pessimistisch): nicht-serialisierbare Schedules werden verhindert, v.a. durch Sperrverfahren (selten Zeitstempelverfahren)

Wie funktioniert das Sperren bei Datenbanken?

Verwaltung der Sperren durch den Lock-Manager: Lock a=Objekt wird angefordert; unlock a= Objekt wird freigegeben. Alle Locks werden in einer Sperrtabelle eingetragen.

- Sperrprotokoll: Wie müssen Sperren gesetzt und freigegeben werden, damit serialisierbare Schedules entstehen? Zwei-Phasen-Sperrprotokoll
- Sperrmodi - Welche Arten von Sperren benötigt man?
  Shared-Lock, Exclusive-Lock, Intendierter Lock, 'kurze Sperre'
- Welches sind die sperrbaren Einheiten in einer Datenbank?
  Databaselock, Tablelock, Datasetlock darüber hinaus noch Arealock um Teile einer Datenbank zu Sperren
- Sperrhierarchie - Datenbank - Area - Tabelle - Tupel/Seite (Lock-Escalation)
- Deadlock-Behandlung (tritt bei exklusiven Sperren auf) wobei Schedules wechselweise auf Freigabe einer Sperreinheit warten
- Livelock: durch wiederholt ungünstige Zuteilung von Sperren kann eine Transaktion permanent blockiert werden.

Wie ist das Deadlock-Problem bei Transaktionen zu behandeln?

- Verhinderung: Preclaiming, Zeitmarken auf Transaktionen (Timeouts)
- Erkennung: Timeout und Prüfung des Wartegraphen auf Schlingenfreiheit oder Kombination beider Verfahren (Aufbau des Wartegraphen bei Timeout)

Erläutern sie die Deadlock-Problematik!

Deadlock: zyklisches Warten durch exklusive Zugriffs-Anforderung (exklusive Locks) mehrerer Schedules auf die gleichen Objekte.

Wie erkennt man Deadlocks und löst sie auf?

Prüfung auf Zyklen im Warte-Graph; gerichtete Kante von Ti nach Tj, wenn Ti auf Tj wartet. Auflösung: billigste Transaktion wird zurückgesetzt.

Wann prüft man auf Deadlocks?

Prüfung auf Zyklen im Wartegraphen: nach Einfügen einer Kante, in bestimmten Intervallen, wenn eine Transaktion über eine gewisse Zeit gewartet hat (Timeout)

Was ist der Unterschied zwischen Deadlock und Timeout?

Timeout: eine bestimmte Transaktion wartet auf ein Objekt länger als eine bestimmte Maximalzeit, trotzdem kann eine korrekte Fortsetzung möglich sein.

Deadlock: mindestens zwei Transaktionen warten jeweils auf die andere => unendliches Warten. Auflösung ist nur durch Abbruch einer Transaktion möglich.

Man erläutere das 2-Phasen-Sperrprotokoll!

Nach dem ersten Unlock eines Transaktionssystems werden von dem TS keine weiteren Sperren mehr angefordert. Die Anzahl der Locks durchlaufen also eine Wachstumsphase und eine Schrumpfungsphase, ein solches Transaktionssystems ist serialisierbar.

KE6 - Transaktionsmanagement II, Integrität

Wie sehen optimistische Verfahren aus?

- Arbeitsphase: nur lesender Zugriff auf die Datenbank (=> kein Rollback nötig), Zwischenspeicherung in privatem Arbeitsbereich
- Validationsphase: Prüfung auf Nicht-Serialisierbarkeit der entstandenen Schedule ó Abhängigkeitsgraph hat Zyklen
  Definition der Abhängigkeit: (S: Schedule (=Folge von Lese und Schreiboperationen); T:Transaktionen in S; Konflikt zwischen zwei Operationen besteht wenn: Zugriff auf dasselbe Objekt, davon mindestens ein Schreibzugriff, T_y ist abhängig von T_x gdw. es Operationen a_x und b_y gibt die in Konflikt miteinander stehen und ax vor by in S ausgeführt werden soll. Der Abhängigkeitsgraph ist dann G(S)=(T,U); (T_y,T_x)Î Uó T_y abhängig von T_x
- Schreibphase: veränderte Objekte werden in die Datenbank geschrieben => Commit.

Was genau passiert in der Validationsphase, welche Operationen werden hier betrachtet?

Es genügt schreibende Zugriffe parallel ablaufender Transaktionen zu betrachten und zu vergleichen mit den Lese- und Schreiboperationen der Transaktion für die die Validation durchgeführt wird. Prüfung auf Zyklen im Abhängigkeitsgraph (Knoten: Transaktionen; gerichtete Kanten zu jeweils abhängigen Transaktion ; Abhängigkeit: min. eine Operation steht im Konflikt mit einer vorher in der Schedule stehenden Operation einer anderen Transaktion)

Wie kann man Transaktionen synchronisieren?

Durch den Einsatz von Synchronisationsverfahren:

- verifizierende Verfahren (optimistisch)
- präventive Verfahren (pessimistisch)
  z.B. Sperrverfahren, Zeitstempelverfahren

Warum benutzt man bei Datenbanken keine Semaphore?

Semaphore stellen bei Betriebssystemen den exklusiven Zugriff auf Betriebsmittel durch Prozesse sicher, dies garantiert im Falle der DB-Transaktionen aber keine Serialisierbarkeit! Exklusive Sperren machen Deadlocks erst möglich (siehe Deadlockvoraussetzungen bei Betriebssystemen)!

Wann ist eine Transaktion rücksetzbar?

- Rücksetzbarkeitsbedingung für Lesen: Transaktion T kann erst dann abgeschlossen werden, wenn alle Transaktionen, von denen T Daten gelesen hat, abgeschlossen sind.
- Rücksetzbarkeitsbedingung für Schreiben: Eine Transaktion darf ein Objekt erst schreiben, wenn alle Transaktionen, die x zuvor geschrieben haben, abgeschlossen oder abgebrochen sind.
- strikte Ausführung: Transaktion T darf ein Objekt x erst schreiben oder lesen wenn alle Transaktionen, die x geschrieben haben, abgeschlossen oder abgebrochen sind.

Wie kann man Deadlocks erkennen und behandeln?

Erkennung durch Prüfung des Wartegraphen auf Schlingenfreiheit, Behandlung durch Abbrechen der billigsten Transaktion bzw. durch den Einsatz von präventiven Synchronisationsverfahren.

Warum ist Deadlockvermeidung unrealistisch?

Bei Produktionsdatenbanken hat man es mit einer großen Zahl sich fortlaufend ändernder Objekte zu tun. Zwischen diesen Objekten bestehen nahezu beliebig komplexe Querbezüge. Die Anzahl der benötigten Sperren ist in der Regel unbekannt. Beim Preclaiming wird zudem die Parallelität der Transaktionen und damit die Gesamtperformanz eingeschränkt.

Was ist der Unterschied zwischen pessimistischen und optimistischen Verfahren?

Pessimistische Verfahren garantieren die Serialisierbarkeit von Transaktionen, optimistische Verfahren stellen die Parallelisierung in den Vordergrund und brechen beim Eintreten eines Deadlocks eine der beteiligten Transaktionen ab.

Wie arbeiten optimistische Verfahren?

Abhängigkeitsgraph bei Commit auf Serialisierbarkeit (=Zyklusfreiheit) prüfen. Falls Zyklen auftreten, Transaktion abbrechen

Können bei optimistischen Verfahren Deadlocks entstehen?

Nein! Ohne Sperren kein Deadlocks!

Bei welchen Synchronisationsverfahren können Deadlocks auftreten?

Sperrverfahren, wenn man 2-Phasen-Sperrprotokoll oder Sperren bis EOT verwendet.

Wie geht das Rücksetzen einer Transaktion vonstatten?

Rücksetzen (Undo): Schreiben der before-Images in umgekehrter Reihenfolge; bei optimistischen Verfahren nicht nötig, da Schreiben erst nach der Validationsphase. Im Log befindet sich ein Mitschnitt der Aktivitäten der Transaktionen.

Im physischen Log werden die Werte von Datenbankobjekten vor und nach Änderungen durch Transaktionen gespeichert (before-/after Image); im Log wird eine Marke für BOT sowie EOT gesetzt. Für ein etwaiges Rollback vor EOT wird ein before Image im Log gesichert. Ebenso wird unmittelbar vor EOT noch ein after-Image der Transaktion ins Log geschrieben. Dies dient zum Recovery nach Speicherfehlern und zum Wiederholen von nach dem letzten Checkpoint beendeten Transaktionen z.B. nach Systemzusammenbrüchen

Im logischen Log werden die durchgeführten Operationen gespeichert.

Welche Transaktion wird bei einem Deadlock zurückgesetzt?

Die billigste Transaktion wird zurückgesetzt.

Kosten stellen hierbei die Anzahl der erforderlichen DB-Zugriffe, die CPU-Zeit und die Anzahl der abzubrechenden Transaktionen (Rollbacks) dar.

Also sollte diejenige Transaktion mit der geringsten Anzahl an Datenänderungen/-selektionen und mit der geringsten CPU-Intensität zurückgesetzt werden!

Was passiert bei einem Undo?

Bisherige Änderungen durch die Transaktion werden dadurch rückgängig gemacht, daß aus dem Logfile das before-Image in die Datenbank zurückgeschrieben wird. Beim Undo wird das Log-File rückwärts durchlaufen (bei einem Redo dagegen vorwärts); ggf. wird ein fortgepflanzter Rollback angestoßen, da durch das Rücksetzen einer Transaktion alle von ihr abhängigen Transaktionen auch zurückgesetzt werden müssen.

Welche beiden Nachrichten enthält implizit die Ready-To-Commit-Meldung einer Subtransaktion?

Es ist jederzeit ein Undo oder ein Redo möglich.

Erläuterung des Recovery bei einem Systemstart (nach einem Absturz):

Sei TC die Menge der zum Zeitpunkt eines Systemabsturzes aktiven Transaktionen.

- Undo/Redo: Durchsuche das Log vom Abbruch bis zum letzten Checkpoint, führe ein Undo aus für die in dieser Zeit nicht abgeschlossenen Transaktionen, für die übrigen Transaktionen aus TC ein Redo
- Undo/No-Reado: physikalisches Schreiben vor dem Commit (schlecht bei Hot-Spots), für die seit dem letzten Checkpoint nicht abgeschlossenen Transaktionen wird ein Undo ausgeführt,
- No-Undo/Redo: nur abgeschlossene Objekte werden in die Datenbank geschrieben, für die seit dem letzten Checkpoint nicht abgeschlossenen Transaktionen wird ein Redo durchgeführt.
- No-Undo/No-Redo: Alle Änderungen werden vor Commit in einer einzigen atomaren Operation die Datenbank geschrieben. Damit dies performant machbar ist, wird Shadowing angewendet. Nachteil: Jeder Zugriff auf das DBS ist indirekt, durch das Anfertigen von Kopien (Shadow) werden die Daten in schwer zu kontrollierender Weise über die Hintergrundspeicher verteilt.

Wie lange muß man before und after-images aufheben?

Before-Images werden für das Rollback von Transaktionen im Falle von

Transaktionsfehlern oder Systemzusammenbrüchen benötigt, Speicherung von BOT bis EOT
After-Images: werden für Recovery nach Speicherfehlern und für das Wiederholen von nach dem letzten Checkpoint beendeten Transaktionen im Falle eines System-Zusammenbruches benötigt. Speicherung unmittelbar vor EOT bis zum nächsten Dump / Checkpoint.

KE7 - neuere Entwicklungen