Ich: (Schluck!, wußte natürlich, wie der funktioniert, aber ich sollte mal ein Beispiel hinschreiben!). Erst mal eingeordnet (Prevention, **Avoidance**, Detection & Recovery). Dann angefangen. Die Vektoren erläutert und versucht alles hübsch aufzumalen. Das hat natürlich nicht hingehauen, so daß ich dazu übergegangen bin die Geschichte nur noch mündlich zu erläutern...

Jarke: Schlägt den Bogen zu Transaktionen und Serialisierbarkeit

Ich: Erzähle was über das Transaktionssystem (r,w,c,a), die Aufgaben des Schedulers und die möglichen Probleme (Dirty-Read, Lost Update, Phantom-Problem)

Jarke: Wie arbeitet denn ein Scheduler? Ähh, ein sperrender!

Ich: erzähle was über 2PL...

Jarke: Kann es denn da Deadlocks geben?

Ich: Jaaa, male ihm ein Schedule hin (zwei Transaktionen, die gegenseitig auf sich warten)

Jarke: Wie kann ich denn erkennen, das Ihr Schedule nicht in CSR liegt?

Ich: Konfliktgraph hat einen Zykel. Hingemalt.

Jarke: Und wie kann man Deadlock svermeiden?

Ich: Erzähle was über konservatives 2PL.

Jarke: Warum macht man das nicht?

Ich: Starvation! (das wollte er nicht hören) 2. Anlauf: Weil der Scheduler ja nicht weiß welche Sperren im Laufe der Transaktion gebraucht werden wird im Allgemeinen zu viel gesperrt. Das war's!

Jarke: Weiter mit Eigenschaften von Schedules. Da gäbe es neben den Klassen der Serialisierbarkeit doch noch etwas ...

Ich: Korrektheitskriterien. PCA, RC (erläutert), ACA (erläutert) ...

Jarke: ... und dann gibt es noch eine Eigenschaft die mit CSR in eine Menge fällt ...

Ich: RG! Rigorose Schedules. (Erläutert)

Jarke: Können Sie denn begründen warum Schedules aus RG auch CSR sind?

Ich: Ja klar. Da alle Konflikte "verboten" sind, ist die Konflikt-Relation immer leer.

Jarke: Sie erwähnten RC. Wie ist denn das mit Recovery? Es gäbe da verschiedene Arten ...

Ich: Die vier möglichen Recovery-Arten erläutert

Jarke: Wann schreibt man denn was in das Log?

Ich: erläutert.

Jarke: Es gibt da so ein Bild. Erläutern Sie mal genau was nach einem Crash so gemacht werden muß.

Ich: Weiß nicht welches Bild er meint. Fange an zu erzählen und erinnere mich an das Bild: Bereits abgeschlossene Transaktionen und welche, die zum Zeitpunkt des Crashs noch laufen. Hingemalt, erläutert. (Es schloß sich noch eine kleine Diskussion an ...)

Jarke: Wir hatten noch gar nichts zur Einführung. Was sind denn FDs?

Ich: Umgangssprachlich erläutert.

Jarke: Wenn ich nun eine Menge Attribute und eine Menge FDs habe, was mache ich dann?

Ich: Normalisierung! Angefangen die NFen zu erläutern

Jarke: Ich glaube Ihnen mal, daß Sie die NFen kennen. Schreiben Sie doch mal die Definition einer FD formal hin!

Ich: (Shit!) Jetzt wird es richtig schlecht. Versuche es mit Hilfe, komme vollkommen durcheinender und gebe schließlich auf (dabei ist es sooo einfach).

Jarke: Wenn das formal nicht klappt, vielleicht können Sie es ja praktisch (ich: ???). Nehmen Sie mal an, Sie hätten eine alte Personaldatenbank und wollen wissen, ob der Name als Schlüssel taugt. Was machen Sie dann?

Ich: Ich checke, ob jeder Name nur einmal vorkommt!

Jarke: Jaaa ... schreiben Sie das doch mal in SQL hin!

Ich: (Grübel, denk, laber:) Ja da gebe ich doch einfach mal alle Namen aus, die mehr als einmal vorkommen. Wenn es solche gibt taugt das nicht als Schlüssel:
SELECT Name
FROM Personal (einfach angenommen ...)
(Hmm, brauch kein WHERE)
GROUP BY Name HAVING COUNT(Name)$\geq$2
(Glücklicherweise hatte ich mir SQL gut angeschaut, so daß das recht locker aus der Hand ging...)

Das war's dann auch — 60 Min vergingen wie im Flug.

3 Bewertung

Prof. Jarke bescheinigte mir einen sehr guten Überblick über die Materie. In der Prüfung war mir jedoch bereits klar, daß ich für seinen Geschmack zu wenig aufs

Papier gebracht habe. Er meinte dann auch, daß für eine Eins auch die Formalismen da sein müßten.

Die Bewertung war meiner Meinnug nach fair und dem Fach angemessen.

4 Atmosphäre, Sonstiges

Die Atmosphäre war angenehm locker, viel besser als ich erwartet hätte.

Ich war im übrigen sehr überrascht, daß so wenig zur Einführung kam. Auch andere große Themenkomplexe wurden überhaupt nicht angesprochen, wie z.B. Optimierung, Index-Strukturen, RA(!) und Join-Implementierung.

Viel Erfolg!

Prüfungsprotokoll : Prakt. Informatik bei Prof. Jarke

Prüfungstermin: 20.04.95

Datum des Protokolls: 20.04.95

Fächer: 1. Einführung in Datenbanken (Vorlesung von Prof. Jarke)
2. Compilerbau (Vorlesung von Prof. Indermark)
3. Betriebssysteme (Buch von Silberschatz: *Operating System Concepts*)

Dauer: ca. 55 min

Note: 1,7

I. Einführung in Datenbanken

Frage: *Was können Sie zum relationalen Datenmodell sagen ?*

Ich erkläre ihm, daß jedes Datenmodell aus Strukturen, Operationen und Constraints besteht. Als Struktur erwähne ich die Relation, worauf er sofort fragt:

Wie ist denn eine Relation definiert ?

Ich erwähne und erkläre die Begriffe *Relationenschema* und *erweitertes Relationenschema*, was ihm offenbar genügt.

Frage: *Warum ist der Begriff Relation gerechtfertigt ?*

Ich erkläre, daß eine Relation eine Teilmenge des kartesischen Produktes der Wertebereiche seiner Attribute ist.

Frage: *Welche Operationen gibt es denn ?*

Ich zähle ihm die mengentheoretischen Operationen auf und erwähne, daß bei der Anwendung der Operationen bestimmte Bedingungen an die Relationen zu stellen sind, beispielsweise die Vereinigungskompatibilität für die Vereinigung , den Durchschnitt und die Differenz. Darauf Prof. Jarke:

Und die speziellen Operatoren ?

Ich zähle ihm die vier (Selektion, Projektion, Join und Division) auf.
Jarke erzählt darauf kurz etwas über die Implementierung dieser Operatoren, und ich bin froh, darauf nichts sagen zu müssen. Dann sagt Prof. Jarke, daß es bei manchen Anwendungen aus Effiziengründen sinnvoll ist, bestimmte Anfragen zu unterstützen.

Wie kann man dies denn tun ?

Mir fallen dazu Primär- und Sekundärzugriffsverfahren ein.

Frage: *Welche Primärzugriffsverfahren gibt es ?*

Ich nenne ihm den (physisch) sequentiellen, den Index-, den indexsequentiellen und den Hash-Ansatz.

Frage: *Welchen gibt es denn noch ?*

Er läßt mich darauf einige Zeit überlegen und sagt dann selbst: *B-Stern-Bäume.*

Frage: *Welchen Unterschied gibt es denn zwischen Index-Verfahren und B-Stern-Baum, da die beiden doch eigentlich gleich sind ?*

Mit Mühe und Not komme ich darauf, daß bei einem zu großen Index dieser nicht mehr in den Hauptspeicher paßt und dann B-Stern-Bäume verwendet werden.

Er will dann wissen, wie ein B-Stern-Baum aussieht. Dies bekomme ich nur mit einiger Mühe hin, da ich mich hierauf nicht besonders gut vorbereitet habe, was wohl ein Fehler war. Anscheinend merkt Prof. Jarke, daß ich hier nicht besonders gut Bescheid wußte, was ihn dazu veranlaßt, mich damit weiter zu beschäftigen: Er gibt mir an, daß es N Datensätze gibt, ein Datensatz d Bytes, ein Pointer p Bytes benötigt und in einen Knoten h Bytes passen. Dann will er wissen, wieviel Datensätze in einen Knoten passen.
Ich schreibe: $k = h / (d+p)$ und erkläre, daß ich dabei noch einen Zeiger vergessen habe, da es immer einen Pointer mehr als Datensätze gibt. Dies ist ihm aber egal.

Frage: *Wieviel Zugriffe benötige ich denn für einen Zugriff ?*

Meine Antwort lautet: $\log N$ (zur Basis k).

Frage: *Ist dies die durchschnittliche, minimale oder maximale Anzahl ?*

Ich behaupte, daß es die (minimale oder maximale; weiß ich nicht mehr) Anzahl ist, worauf Jarke sagt, daß es immer die genaue Anzahl ist.
Schließlich soll ich noch einen B-Stern Baum aufbauen, indem ich sukzessive die Zahlen 1,...,7 einfüge. Da ich von B-Stern-Bäumen eigentlich keine Ahnung habe, geht dies ziemlich schief, worauf Jarke mit B-Stern-Bäumen endlich aufhört.

Zu meinem Erstaunen fragte Prof. Jarke nichts zum ER-Modell, Hierarchischen- oder Netzwerk-Modell, nichts zum Tupel oder Domänenkalkül, nichts zu SQL und auch nichts zur Dekomposition, Normalisierung oder Synthese.

Gedächtnissprotokoll

Was: Praktische Informatik
Wann: 8. Dezember 1994
Bei wem: Prof. Jarke
Dauer: 45 Minuten
Note: 2.3

Fächer: Betriebsysteme (Silberschatz),
Einführung in Datenbanken (Jarke),
Implementierung von DBS (Jarke)

Zusätzliche Literatur:
Datenbank-Handbuch
Herausgeber: P.C. Lockemann und J.W. Schmidt
Springer Verlag

Betriebssysteme:

Wir haben uns sehr ausführlich über die Prozeßzustände und Übergänge unterhalten (ready, waiting, blocked). Hier nun einige Stichworte die gefallen sind:
Wo welche Queue ?, Interrupts, wie werden die ausgelöst/verarbeitet ?, Scheduling-Algorithmen, dual mode, PCB, Auslagern/einlagern von Daten, Register retten, ...
Bis hierhin hatte ich immer alles aus der Sicht eines Prozesses beschrieben. Er wollte das aber aus der Sicht des Prozessors. Wie erfolgt den nun tatsächlich die Umschaltung zwischen den Prozessen ? Das BS ist ja in dem Fall selber ein Prozeß das erstmal in den Prozessor eingelagert werden muß. -> unteilbarer Befehl, ... Wie wird der neue Prozeß gestartet ? Befehlszähler an die richtige Stelle setzen, ...
Dann kamen wir zum virtuellen Speicher. Stichworte: page fault und was dann alles passiert, welche Speicherstrukturen werden verwendet ? Wie Verwaltet ? ...

Einführung in DB und Implementierung von DBS:

Über den virtuellen Speicher sind wir dann zum Schichtenmodell übergegangen. Hier habe ich dann etwas über die verschiedenen Schichten erzählt. Insbesondere über die Systempufferschicht.
Dann Transaktionsverwaltung: 2PL-Scheduler, warum korrekt ?, final-state-, view-, commit-serialisierbar, Konflikt-Graph -> keine Zyklen, ...
Weitere Eigenschaften von TA'en: rücksetzbar, Vermeidung kaskadierender Aborts, Rigorose Schedules, ...
Wie vermeidet man Deadlocks ? 2PL, Zeitmarken, Serialisationsgraphentester, ...
Dann sind wir zu Dekomposition und Synthese gekommen: Normalisierung 1-3, BCNF, Anomalien bei Redundanz, ...
Warum Syntheseverfahren ? FD-Erhaltend, ...
Wie kann man den Informationsgehalt bei Dekomposition erhalten ? Armstrong-Axiome, ...
Wie kann man die beweisen ? Direkt aus den FD's.
Können sie das beweisen ? Nö.
Auch gut.

Sehr angenehme Atmosphäre. Prof. Jarke ist sehr ruhig und überhaupt nicht chaotisch während der Prüfung (im Gegensatz zu den Vorlesungen). Er stellt keine unangenehmen Fragen und falls es mal doch nicht sofort klappt, gibt er gute Tips.

Viel Glück

Diplomprüfung praktische Informatik

Prof. Jarke, 12. Januar 1993

Themen: Betriebssysteme, Datenbanken, Implementierung von Datenbanken

Note: 2.0
Literatur : Silberschatz, Peterson (3. Auflage) Teil 1 - 3 und UNIX-Teil
Datenbanken-Vorlesung (Prof. Walter)
Datenbankhandbuch Kapitel 3 und 4
Vossen, Datenmodelle
(leider keine Prüfungsprotokolle)

Welche Strukturen, Operationen, Constraints hat relationales Datenmodell ?
1 Strukturart : Relationen; Tabelle: Spalten = Attribute; Zeilen = Tupel
Beispiel-Relation angeben
(Bei Erklärung Normalform erwähnt.)

Op < rel. Algebra - prozedural
rel. Kalkül - deskl.

Welche Normalformen gibt es, wozu NF?
1. NF nur atomare Attribute
2. NF jedes Nicht-Schlüsselattribut ist funktional abhängig vom Promärschlüssel
3. NF kein Nicht-Schlüsselattribut ist transitiv abhängig vom primär Schlüssel
Boye-Codd-NF alle funktionalen Abh. hängen vom Primärschlüssel ab

Definition Primärschlüssel und funktioal abhängig?

Gibt es nur einen Primärschlüssel ?
mehrere

Warum werden NF angewandt ?
Aufhebung von Redundanzen und Anomalien
Nachteil : Dekomposition => erhöhte Zugriffszeiten

Wie normalisiert man eine Relation ?
1. NF -> 2. NF teile Relation auf Basis der Teilschlüssel-Abh.
2. NF -> 3. NF aufteilen der Attr. in der Mitte der Transitivität

Verlustfreie Joins und Abhängigkeitserhaltung erklären !
Boyce-Codd nur verlustfrei nicht abh.erhaltend

Dekomposition und Synthese erklären !

Operationen des rel. Datenmodells !
relationale Algebra und relationales Kalkül
2 Arten von Operatoren des rel. Modells: mengentheoretische und relationale
relationale Op.: Selektion, Projektion, Join, Division

Op < rel. Algebra - Prozedural → mengentheoretische
→ rel. Op.
rel. Kalkül

Prüfungsprotokoll Praktische Informatik

Prüfer	:	Prof. Dr. Matthias Jarke	
Datum	:	8. Dezember 1994	
Fächer	:	1. Einführung Datenbanken	(Jarke)
		2. Implementierung von Datenbanken	(Jarke)
		3. Einführung in die Softwaretechnik	(Nagl)
Dauer	:	ca. 1 Std.	

Softwaretechnik wird nicht als Block abgefragt, sondern zwischendurch immer mal wieder eingeschoben.

- ER-Diagramme:
  Entwerfe ER-Diagramm für:
  - Autor mit Adresse
  - Buch mit Buchtitel und Seitenzahl
  - Ein Autor kann mehrere Bücher schreiben
  - Ein Buch kann mehrere Autoren haben

  (Mit zwei Entitäten und einer Relationship modelliert)

- Vorgehensweise bei Realisierung eines Softwareprojekts:
  - Istanlsyse
  - Problemanalyse
  - ... (s. Skript)

- In welche Phase gehört ER-Diagramm?
  Anforderungsdefinition

- Womit erstellt man Spezifikation?
  SA (Structured Analasys)

- Was ist SA?
  - Prozessoren
  - Datenflüsse
  - Terminatoren
  - Datenspeicher

- Welche Regeln müssen bei Modellierung mit SA eingehalten werden?
  - Keine unbenannten Datenflüsse
  - Keine Prozessoren, in die nur Datenflüsse nur hinein oder nur herausführen
  - ... (s. Skript)

- Wie geht es dann weiter (Verfeinerung)?
  - Minispecs
  - DataDictionary

- Modellierung des ER-Diagramms als Netzwerkmodell:
  Aufspaltung der m:n-Beziehung in 2 1:n-Beziehungen durch Einführung eines Zwischenrecords

- Wie wird sowas denn implementiert?
  Multilisten

- Modellierung im relationalen Modell:
  Pro Entität und Relationship eine Relation ==> drei Relationen

- SQL-Anfrage:
  Name der Autoren, die ein Buch mit mehr als 100 Seiten geschrieben haben, und die Anzahl der Bücher, die mehr als 100 Seiten besitzen, nach Autoren geordnet.

```
SELECT    Name, COUNT(Buch)
FROM      Schreibt, Buch
WHERE     Seitenzahl > 100
GROUP BY  Name
```

- Relationenalgebra:
  GROUP BY und COUNT weg. Gleiche Anfrage jetzt in Relationenalgebra

- Implementation des Joins:
  - Nested Loop
  - Sort/Merge
  - Hash-Joins

- Was ist ein Semi-Join?
  Projektion des Joins zweier Relationen auf die Attribute der ersten Relation

- Wie modelliert man auf Architekturebene?
  Notation von Nagl mit Modulen erklärt. Gehört allerdings nicht in die Einführung, sondern in Spezialvorlesung "Methodisches Programmieren im Großen" von Nagl.

- Was gibt es für Testverfahren?
  - Blackbox-Testen
  - Whitebox-Testen

- Schichtenarchitektur von DB:
  5 Schichtenarchitektur mit ihren Schnittstellen erklärt

- Transaktionsverwaltung:
  - Was heißt serialisierbar?
  - Wozu braucht man das überhaupt?
  - Welche Arten gibt es? (Final-State-, View-, ... Serialisierbarkeit)

- Konfliktserialisierbarkeit:
  Definition angegeben

- Beispiel für einen Konflikt:
  - Dirty-Read
  - Lost-Update
  - Phantom-Problem

- Welche Arten von Schedulern gibt es?
  - sperrende und
  - nicht-sperrende Scheduler

- 2-Phasenprotokoll ==> Zeit ist vorbei

Gedächtnissprotokoll Prüfung: Praktische Informatik (Jarke)
Datum: 11.12.2001
Dauer: 45 Minuten
Note: 3.3
Fachkombination: Einführung in Datenbanken
Implementierung von Datenbanken
Datenkommunikation I + II
Vorbereitungsunterlagen: Vorlesungsfolien
Computernetzwerke (Tanenbaum)
„Der Vossen"
Jarke-Prüfungsprotokolle

Fazit: Als Prüfer sehr zu empfehlen!!!

Implementierung von Datenbanken (ca. 20 Minuten

- Transaktionsverwaltung
- Read-Write-Modell
- Lost-Update-, Dirty-Read- und Phantom-Problem
- Serialisierung
- FSR, VSR, CSR
- Konflikt- Graph
- Fehlerrecovery
- ST, ACA, RC
- Scheduler (2PL)
- 2PC

und Übergang zur Datenkommunikation!

Datenkommunikation (ca. 10 Minuten)

- TCP/IP
- ISO-Schicht 1
  - Physikalische Ebene
- ISO-Schicht 2
  - HDLC, Fehlererkennung und Behebung (u.a. CRC)
- ISO-Schicht 3
  - Routing

Wir müssen ja noch was zu Einführung in DB tun.

Einführung in Datenbanken (ca. 15 Minuten)

- Wie geht man beim DB-Entwurf vor (RE -> ER-Diagramm -> Relationales Modell) eher allgemein behandelt
- Normalformen (1NF, 2NF, 3NF, funktionale Abhängigkeiten, Umformung)

Irgendwie fand ich die Prüfung vor allem im Ablauf echt krass, wenn man sie mit anderen Protokollen vergleicht! Note lag sicherlich an meiner Nervosität und auch daran, dass ich doch noch etwas mehr hätte tun können/sollen. Die Prüfung hätte jedoch auch um einiges leichter sein können, wenn man etwa den Weg ER-Diagramm -> Relationenmodell -> SQL zu Beginn der Prüfung in EDB zuerst als Aufgabe bekommt!

Gedächtnisprotokoll Professor Jarke

Fach	Vertiefungsgebiet: Informationssysteme
Datum	Juni 1994
Dauer	ca. 45 Minuten
Vorlesungen	Einführung in Datenbanken (Jarke) Implementierung von Datenbanken (Kemper) Objekt–orientierte Inf.verwaltung f. ingenierwis. Anwendungen (Kemper) Verteilte Datenbanken & Interoperabilität (Jeusfeld)

Einf.i.DB

Sie kommen in einen Betrieb und sollen eine DB entwerfen, was können sie da machen?

Fragen der zukünftigen Benutzer oder Ansehen schon bestehender Lösungen in diesem oder anderen Unternehmen

Modellierung des folgendes Flugabrechnungszettels als ER-Diagram:

Fluggesellschaft					Datum
Adresse					
Flug#	von	nach	Preis	Anzahl	Einnahmen
...	...	...	...	...	...
					Gesamteinnahmen

Mit 4 Entitäten und 3 Relationships modelliert. (Es geht auch mit nur 3 Entitäten und 2 Rel.ships, wenn man einem Rel.ship ein Attribut anhängt.)

Relationales Schema dazu

(beachte: 4 Relationen reichen, berechnete Attribute, künstliche ID# als Primärschlüssel)

Anfragen auf diesem Schema: Wie berechnet man die Einnahmen pro Flug (also die Spalte Einnahmen in der obigen Tabelle)

*SELECT-Anfrage mit Join und berechnetem Attribut (in der SELECT-Zeile) F.Preis * B.Anzahl hingeschrieben*

Und wenn man nun die Gesamteinnahmen je Fluggesellschaft haben will?

*Wieder ein Join diesmal über 3 Relationen; SELECT-Zeile mit berechnetem Attribut SUM(F.Preis * B.Anzahl) und an das ganze wird noch ein GROUP BY A.Fluggesellschaft angehängt*

Implementierung von DB

Wie implementiert man sowas?

Mit einem DBMS! (Die Frage war mir dann doch etwas allgemein)

Wie genau? Schichtenarchitektur?

oberste Schicht: Mengenorientierte Schicht; Aufbau des Anfragebaums; Optimierung des Anfragebaums mittels Runterziehen von Selektion; Selektion & Kreuzprodukt ⇒ Join, ...

Was passiert mit dem Anfragebaum in tieferen Ebenen

Implementierungsmöglichkeiten des Joins: nested loop; nested loop mit erweitertem Pufferspeicher; nested loop unter Verwendung eines Indexes; Sort–Merge–Join; Hash–Join

Was passiert dabei wo in den tieferen Ebenen?

Erzählt

Wofür eigentlich Segmentschnittstelle & Dateiverwaltungsschnittstelle, wo es sie doch schon im BS Virtual Memory, Cache und Filesystem gibt?

Dateiverwaltungsschnittstelle: außer Portabilität nur Nachteile (Umrechnung Datei → Platten-Block im BS notwendig)

Segementschnittstelle: bessere Information über Inhalt der Seiten; eigene Ersetzungsstrategien

(von der Antwort bin ich nur begrenzt begeistert; fehlt zumindest noch: eigenständige Kontrolle über Puffergröße und Entscheidung über STEAL- und FORCE-Eigenschaften)

Wo im Schichtenmodell liegt die Transaktionsverwaltung?

Parallel zu Ebenen I_2 bis I_5; Gründe dafür

Wie implementiert man denn diese Aufteilung auf verschiedene Schichten?

Wußte ich nichts genaues zu, habe dann was über Mehrschichten-Transaktionen erzählt

Recovery: Arten?

R1 bis R4 jeweils mit Beschreibung

Wie implementiert man R3?

Physische (R3: before-images) und logische Protokollierung; Vor-/Nachteile; Arten von Sicherungspunkten; Verwendbarkeit für R3

Verteilte DB

Was wurde denn in der Vorlesung gemacht?

Äh?!?! Vorteile/Nachteile aufgezählt

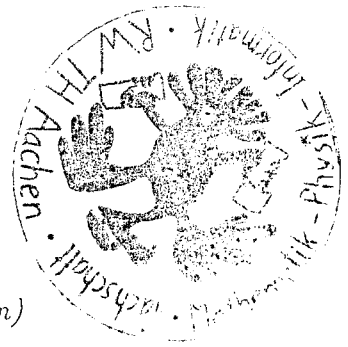
Integration heterogener DB

2 Varianten bei vert. DB:

1) globales Schema gegeben ⇒ Fragmentierung ⇒ Allokation

2) lokale Schemata gegeben, globales gesucht; (← heterogene Datenbanken)

Vorgehen und Probleme: siehe Manfreds Vorlesung



Was kann man machen, wenn man aufgrund der Probleme kein globales Schema erhält?

??? Was über Interoperabilität u.ä erzählt; DB Ansatz, PS Ansatz

(Wer weiß was er da wirklich hören wollte)

OODB

Also ich als Anwender war da gerade auf einem Kongreß, wo mir erzählt wurde, OODB wären was ganz tolles. Warum haben sie das Flugabbrechnungsbeispiel nicht objektorientiert modelliert?

Vor-/Nachteile aufgezählt; bei kaufmännischen Anwendungen eher nicht sinnvoll

Modellierung des Ganzen (Er wollte eine Definition sehen)

GOM Modellierung als <u>ein</u> Objekt (auf ausdrücklichen Wunsch Jarkes; normalerweise furchtbare Anomalitäten)

er wollte wohl hauptsächlich auf die Methoden für die berechneten Attribute heraus, das habe ich nur nicht mitgekriegt

Allgemein: Wahrscheinlich ist es nicht so sonderlich vorteilhaft, wenn man von den 4 Vorlesungen über die man sich prüfen läßt, nur eine beim Prüfer gehört hat. Der Prüfer weiß nicht so genau was er fragen soll, und man selbst weiß nicht, worauf er eigentlich hinaus will.

von 'älteren' Transaktionen während jüngere Transaktionen warten müssen oder zurückgesetzt werden. Ein ähnlicher Mechanismus wird auch für die Serialisierung von Transaktionen eingesetzt.

Prüfungsprotokoll - praktische Informatik

Tilmann F. W. Bruckhaus

1 The facts

- Prüfer: Matthias Jarke
- Bereich: praktische Informatik
- Fächer:
  - Compilerbau
  - Betriebssysteme
  - Datenbanken
- Prüfling: Tilmann Bruckhaus
- Note: 1.3
- Vorbereitung: ca. 3 Wochen
- Datum der Prüfung: 10.12.1991
- Datum des Protokolls: 12.6.1992 (sorry)

2 Material

- Compilerbau: Vorlesung Klaus Indermark (und sonst gar nichts)
- Betriebssysteme: das Buch vom Silberschatz. (ca. die ersten 300 Seiten)
- Datenbanken: das andere Buch vom Silberschatz (ca. die ersten 300 Seiten + das Kapitel über Unix) + Jarki's Vorlesung

3 Anmerkung

Jarkis db Vorlesung war natürlich Prüfungsrelevant, aber da ich in dem Semester in dem Jarki seine Vorlesung hielt meine Diplomarbeit geschrieben habe, habe ich die Vorlesung selten besucht und auch nicht viel in die Folienkopien gesehen.

Ich habe mich also fast ausschliesslich anhand von 2 Büchern und einer guten Vorlesungsmitschrift eines Freundes vorbereitet.

Noch 'ne Anmerkung: Ich habe nach meinen Prüfungen fluchtartig das Land verlassen und mache jetzt hier in Canada einen PhD. Folglich basiert das Folgende auf blassen Erinnerungen aus einer anderen

Welt. Dieses Protokoll entstand aus einer email, die ich einem Kumpel geschrieben habe. Der hat diese Woche Prüfung bei Jarki und wollte von mir wissen, was der so gefragt hat. Ich dachte mir, dass vielleicht noch andere an meinem Geplapper interessiert sind...

4 Die Prüfung

Ja, also in meiner Prüfung konnte ich mir die Reihenfolge der Fächer aüssuchen. Erst hab ich Compilerbau gewählt, weil ich mir die zwei Hämmer bis zum Ende aufheben wollte.

4.1 Compilerbau

4.1.1 Die Phasen

Da hat er mich gefragt: "Wie ist denn das mit den Phasen: wie macht denn der Compiler das." Ich erzähl: Eingabe / Ausgabe / Hilfsdatenstuktur jeder Phase. Da ist so eine Tabelle am Anfang des Skripts. Als ich bei der drittletzten Phase bin und noch nicht gestottert hatte oder etwas ausgelassen hatte, sagt de Jarki wörtlich: "Ich sehe zu meinem Missvergnügen, dass sie das drauhaben." (Damals verstand er wohl noch Spass. Er hatte erst vor einigen Monaten in Aachen angefangen. Ich habe jedoch kürzlich eine Geschichte aus Deutschland gehört, die darauf hindeutet, dass er im Moment weniger zu Spässen aufgelegt ist.)

4.1.2 Implementation

Dann wollte er was dazu wissen, wie man denn einen Compiler implementiert. Lexx und Yacc und so weiter. Er meinte aber dass es da noch eine andere Methode gäbe. Ich wusste erst nicht was er will. Er hift mir aber indem er sagt: "wenn Sie sich die Produktionen der Grammatik ansehen, was kann man denn mit denen machen?" Und: Klick! "Jaja, der rekursive Abstiegscompiler! Da schreibt man sich rekursiv aufrufende Prozeduren für die Terminals und die Nonterminals." Es gibt da noch einen Unterschied zwischen diesen beiden Klassen von Prozeduren. Den wusste ich damals auch. Ich denke die Prozeduren für die Terminals rufen keine weiteren auf. Das habe ich damals ziemlich genau erklärt. "Dann wollen wir mal zum nächsten Thema kommen. Was darf es den sein?"

4.2 Betriebssysteme

Ich sage ich will - was war noch das zweite Thema? Ist alles so lange her - also: db war das dritte - ähh - ach ja - Betriebssysteme:

4.2.1 Kooridination

Also er will wissen: "Da gibt es doch Mechanismen zur Koordination. Ich also die ganzen Dinger aufgezählt: Monitor und was nicht alles und auch die Unzulänglichkeiten der einzelnen Mechanismen aufgezahelt. Und erklärt wie das im Prinzip funktioniert. Er wollte aber dann nicht die Programme zur Implementation aufgemalt haben vor denen ich ja soviel Angst hatte. Die werden recht kompliziert und ich hatte schon auf einer höheren Abstraktionsebene genug erzählt. Mein Tip: Halte dich mit den Programmen beim lernen nicht zu lange auf. Ich musste auch nicht alles wissen. Einen Mechanismus habe ich nicht geliefert und er hat auch nicht nachgefragt. Ich hab auch nicht die historische Entwicklung gebracht sondern habe erst den erklärt der alles kann und dann ein paar einfachere Mechanismen. Die

Beispiele mit den Philosophen und den chop sticks und dem barber shop und so weiter habe ich, soweit ich mich erinnere, nicht erwähnt. Das hätte aber wohl nicht geschadet.

4.2.2 Deadlocks

Und dann wollte er was zu Totschlössern wissen. Erst mal die 4 Bedingungen, von denen ich nur 3 wusste. Die triviale, dass man Geräte benutzt, die nur von einem Benutzer zu einer Zeit benutzt werden können, fiel mir nicht ein. War aber egal. Ich musste auch erklären was das bedeutet: "Kann man eine Bedingung ausschliessen, so ist kein deadlock möglich". Und auch der Unterschied zwischen erkennenden und verhindernden Mechanismen wurde erklärt. Dann habe ich noch gesagt wie man anhand dessen prinzipiell Totschlösserverhinderungs- oder Reparatur- mechanismen installieren kann.

4.2.3 Paging

Dann kam noch die Frage nach paging. Das paging habe ich ihm verklickert und auch die verschiedenen Mechanismen und den idealen mit der Zukunftknowledge. Was ja leider unmöglich zu implementieren ist. Und wie man das heuristisch annähern kann und mit Vergangenheits- statt Zukunftsdaten. Ich glaube die Analogie zu den Sekundärspeicherverwaltungsmethoden kam nicht.

4.2.4 Unix

Dann kam noch ein bisschen unix - was war das denn noch? Ach ja: wie das bei unix gehandhabt wird. Da habe ich was von swapping aber kein paging in älteren Versionen von unix erzählt - glaube ich.

4.3 Datenbanken

Und dann kam der Praxis-Hammer: die db. [Das war für mich deshalb der Hammer, weil Jarke ein db-Spezialist ist und ich nicht.]

4.3.1 Die drei Ebenen

War aber recht easy: Erst ging es um die drei Ebenen ("Konzeptuel", "Schematisch" und "Sicht" oder so ähnlich) und dann habe ich das Entity-Relationship-Modell (ER) für die konzeptuelle Ebene genannt (nach dem Erklären der Ebenen).

4.3.2 Entwurf

Dann ging es weiter mit den verschiedenen db Techniken ("relational", "Netzwerk" und "hierarchisch"). Ich sollte ein kleines Datenmodell in ER-Notation hinmalen. Jarki hat dann so in den Raum gefragt, in welche db struktur der Prüfling denn das soeben erschaffene ER-Diagramm übersetzen solle. Der Beisitzer meinte: "hierhaschisches Modell". Da lachte der alte Jarki nur und meinte, er würde alles tun dieses Modell so schnell wie möglich vergessen zu machen... Also war erst das relationale Modell dran und das Netzwerkmodell sollte dann später kommen. ...kam aber mangels Zeit nicht mehr.

4.3.3 Attribute und Schlüssel

Ja ich hab also die ER-Konstrukte erklärt und dann ein paar Entities und Relationen gemalt. Und auch die Attribute wollte er nicht ausgespart wissen. Ich wusste nicht was er meinte, als er meinte, dass es ausser den Kästen und Rauten noch anderes auf der Welt gäbe also riet ich: "Frauen!... Bier!...

Generalisierung / Spezialisierung?... Aggregation?" Er schüttelte aber nach jedem dieser Versuche sowohl seinen Kopf als auch sich selbst. Er meinte schliesslich: "Die Attribute!". Die sind in seiner Notation Ovale ausserhalb der Kästen und Kreise anders als bei Nagl. Das wusste ich dann aber und habe einige Ovale gemalt. "a1", "a2" usw. und ich sollte auch noch Schlüssel-Attribute bestimmen. Ich mach das auch treu und brav und er will auch noch einen Schlüssel aus 2 Attributen. Ok. war mir auch recht. Dann kam's:

4.3.4 Schema

"Jetzt bestimmen sie mal die Tabellen in der relationalen db. Till beginnt zu schwitzen. Obwohl das heutzutage jeder Grundschüler in der Mengenlehre-Vorlesung lernt und das auch im Buch und in der Vorlesung von Jarki drinne war hatte ich das immer für sooo einfach gehalten, dass ich diese Kapitel überschlag(f)en hatte. Das sollte man nie tun! Aber da es wirklich einfach war habe ich es nach einer kleinen Bedenkpause von 10 sek. auch gekonnt. Erst war's falsch aber Jarki hat dem wohl keine Bedeutung zugemessen, weil's wirklich einfach war und ich dann sagte: "Ja-ja natürlich da hab' ich mich verschrieben."

Ich hab' dann noch erzählt, wie man die Attribute wiederholt in verschiedenen Tabellen, um die Relationen zu realisieren herzustellen. Um ehrlich zu sein, war mir am Anfang nicht einmal klar, dass man natürlich für jede Relation auch eine Tabelle braucht. Glücklicherweise meinte Jarki nachdem ich die Tabellen für die Entities aufgemalt hatte, dass ich nun die Tabellen für die Relationen aufschreiben solle. In diesen Tabellen fand sich dann auch ein schönes Plätzchen um die Attribute der Relationen unterzubringen. (Wo soll man auch sonst damit hin?)

4.3.5 Optimierung

Jarki hatte sicher gemacht, dass ich ein ER-Modell male in dem einige Schlüssel mehrmals vorkommen. Er wollte noch wissen, wie man das jetzt optimiert. Ich bin etwas hilflos. Ich war so stolz auf meine kleinen Tabellen und fand eigentlich nicht, dass es da noch was zu optimieren gäbe. Na er hilft mir ein bisschen und meint, dass ich mal mein Augenmerk auf die Schlüssel richten solle, die mehrmals vorkommen. Also denke ich, dass man Tabellen zusammenfassen kann, wenn ein Schlüsse einen anderen als Teilschlüssel hat. Dann braucht man den Teilschlüssel nicht zu wiederholen. Ich hielt es für nötig ihn darüber aufzuklären, dass man dann aber nicht so schnell auf die Relationen zugreifen kann, wegen der grösseren tabellen, geil? Ausserdem muss man den ganzen Satz wiederholen, wenn ein Schlüsselwert mehrmals in einer Relation auftaucht und so weiter...

4.3.6 Algebren

Eins hätte ich hätte es fast vergessen. Er wollte auch die Symbole für die algebraischen Operationen wissen. Und die beiden Gruppen derselben. Habe jetzt vergessen wie die heissen. Dann wollte er wissen, wie man die ineinander überführen kann. Vorher noch, ob sie äquivalent in ihrer Aussagekraft sind. Das Überführen klappte in so fern nicht, als ich die Existenz- und All-Quantoren an die falsche Stelle in dem Ausdruck fabriziert habe.

Er wollte noch wissen unter welcher unerlässlichen Bedingung der Join nur klappen kann. Ich nich' wissen und labern. Er nach drei Versuchen gesagt: "Die Schlüssel müssen gleich sein." ich hab' natürlich so getan als wäre diese Voraussetzung so fundamental und trivial, dass ich das nun wirklich vorausgesetzt hätte. Tatsächlich hatte ich bis dahin gedacht, dass die Tabellen gleich viele Spalten haben müssen - wie bei Matrixoperationen, weisste - warum ich das glaubte? - frag mich bitte nicht. Ich hatte wohl