

### 3 Implementierung von Datenbanken

- Na gut, dann fragen wir noch mal schnell was zur Implementierung. (Es waren schon 35 Minuten rum).  
Erzählen sie mir doch mal was zur Joinreihenfolgeoptimierung.
- Ich erzähle ihm also die Story von den möglichen Permutationen (wann will Joins nicht zum kartesischem Produkt entarten lassen) und von Kostenmodellen für Zwischenergebnissegrößen. Optimale Reihenfolge der Zwischenergebnissegrößen. Parameter Systeme. Gültige Parameter Systeme erklärt.
- OK OK. Was gibt es da denn noch ?  
Besonders gut ist wenn man Semijoins erhält. Viel günstigere Auswertung etc.

- Wann kann man eine Anfrage durch Semijoins auswerten ?  
Die Geschichte der guten und der bösen Ausdrücke. Hinreichend für böse: Zykeln im Quantigraph. Notwendig: Nicht als Genereller Semijoinausdruck darstellbar. Noch ein paar Schachtem aufgemalt, einstufige Prädikate daran erklärt.
- Ja. Da gibt es ja noch so was wie Transaktionsverwaltung.  
Ich erzähle von Serialisierbarkeit. Definition hiervon. Die Klassen  $FSR \subset VSR \subset CSR$ . Herbrand-Semantik. LRF, RF Mengen.
- Mit welchem Aufwand ist FSR also zu testen ?  
Exponentiell. Es sind  $n!$  Permutationen (im Worst-case) bei  $n$  zu betrachtenden Transaktionen zu behandeln.

- Wie sieht das noch mit Fehlertoleranz aus ?  
 $RC \subset ACA \subset ST \subset RG$  erklärt und definiert. Da  $RC$  orthogonal zu  $FSR$  ist, muss man beides fordern.
- Wie wird das in der Realität gemacht ?  
Z.B. 2PL Scheduler. Besser S2PL oder SS2PL oder konservatives 2PL (geht meistens nicht). Und noch ein bisschen dazu rumgelabert.
- Gut. Wann haben wir angefangen ? Beisitzer: Vor genau 45min.  
OK das war's.

### 4 Fazit

Aus Protokollen zu Prüfungen über Expertensysteme war ja schon zu entnehmen, daß Jarke DATALOG liebt. Also ganz klar selbst Schuld, die Magic-Set Transformation nicht draufzuhaben. Das war auch die Begründung (+ der eine Hänger EBR → RAIDB) warum es nicht zur 1.0 reichte. Schade, in anderen Vertiefungsprüfungen fragt er halt weniger zum dritten Fach. Ich kann auf jeden Fall bestätigen, was auch schon in anderen Protokollen stand: Jarke ist ein sehr lockerer, angenehmer Prüfer. Kommt man nicht weiter, gibt er Tips. Ich hatte mich zweimal verschrieben (z.B. hatte ich zweimal Group by, statt einmal Group by und Order by in der SQL-Anfrage geschrieben). Jarke sagt beide Male "Ääh, ich kann nicht so genau lesen was da links unten steht". Also ein dezentler Hinweis, daß etwas nicht stimmt.

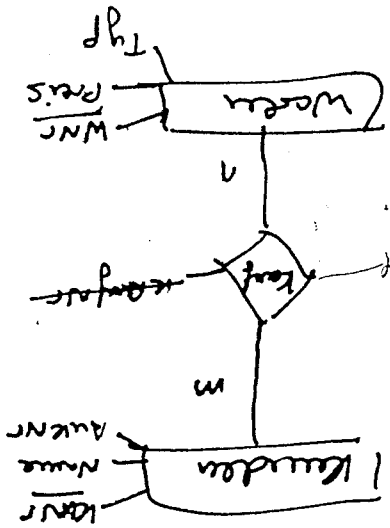


SUPERMARKT

Kunde	Kunde	Name	Name
Kunde	Kunde	Name	Name

Waren	Waren	Preis	Typ
Waren	Waren	Preis	Typ

Kauf	Kauf	Kauf	QTY
Kauf	Kauf	Kauf	QTY



⇒

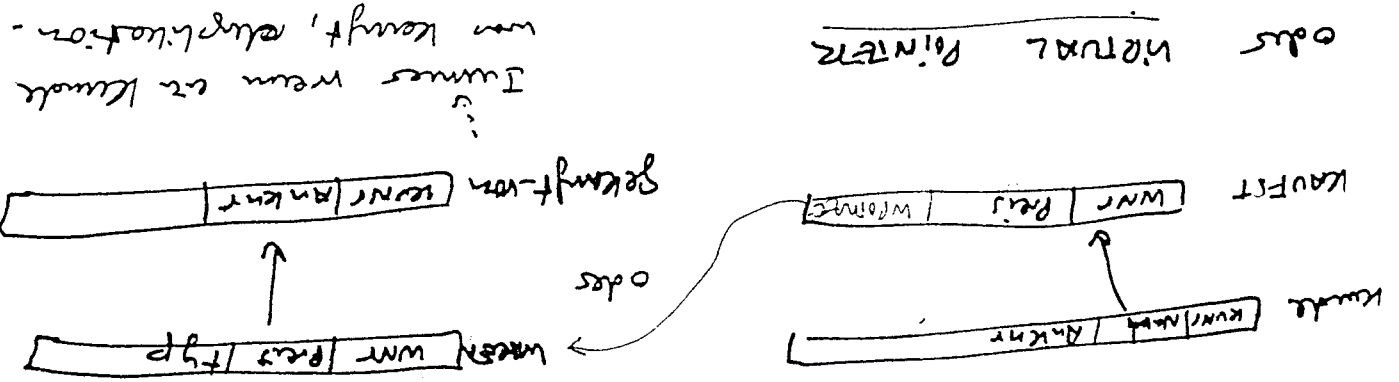
Kunde	Kunde	Name	Name
Kunde	Kunde	Name	Name

Waren	Waren	Preis	Typ
Waren	Waren	Preis	Typ

For each Entity in Record  
 For each Relationship in Set  
 1:1 1:n  
 Attribute der Relation in Menge

HIERARCHISCHE



- WELCHE WAREN EIN KUNDE GEKAUFT HAT.  
 SELECT WARE, LENGE FROM KAUF
- ALLE WAREN UND GEMAUFT VERKAUFT MANGE  
 SELECT SUM(QTY) FROM KAUF  
 GROUP BY WARE

Immer wenn es Kunde  
 von Kauf, Relation.

SELECT COUNT(\*) FROM KAUF  
 GROUP BY WARE

597

SELECT SUM (G.P.RIS)

FROM Behandlung, Rechnung, MASS

WHERE R.BAUM='1990' AND RPNR = P.PNR AND RPNR = M.MASS.

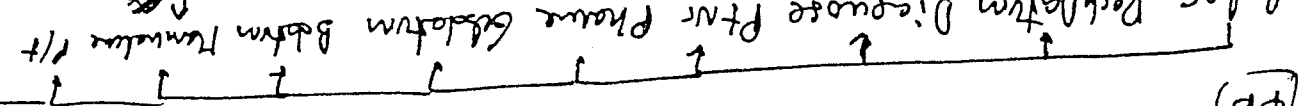
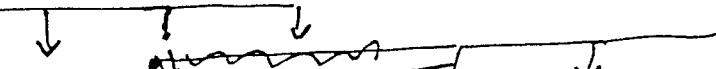
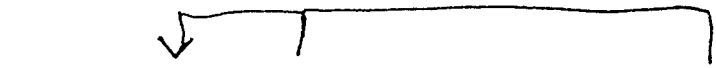
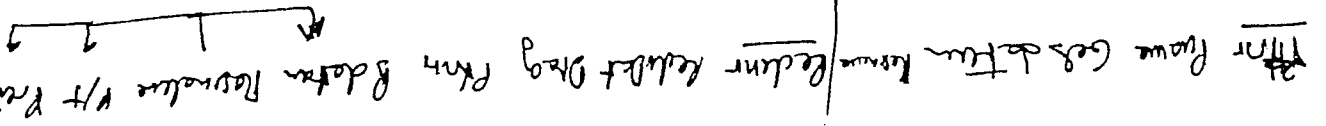
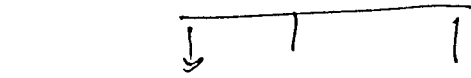
GROUP BY DIAGNOSE AND H.NR AND P.MASS

Personen

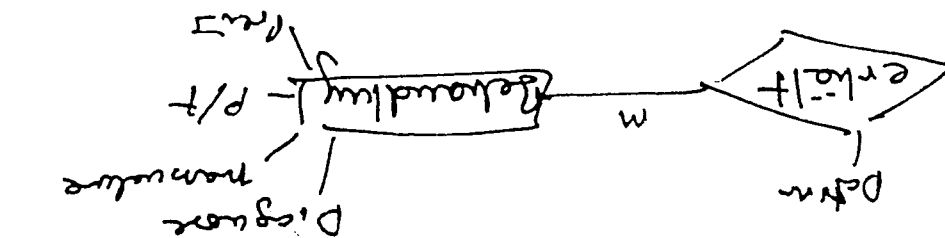
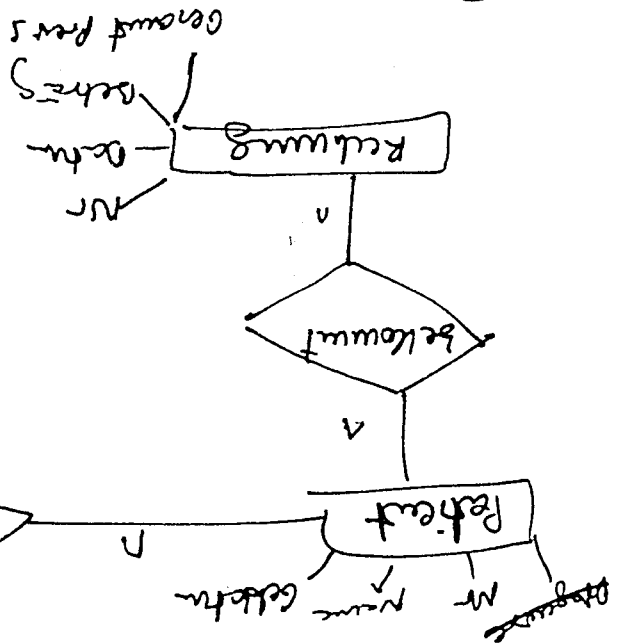
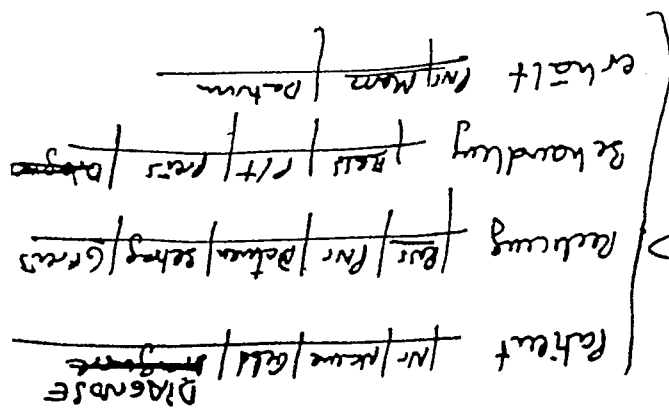
Personen

Personen

Personen



FD



Patient

Koll → Org, ges.

ReduNr  
 Diagnose  
Pnr  
 PatName  
 GebDatum  
 GenusKonten

7 NF

ReduNr  
 Diagnose  
 GenusKonten  
ReduNr  
 PatName  
 GebDatum

2NF

ReduNr  
 PatName  
 GebDatum

2 NF

PatNr  
 PatName  
 GebDatum

ReduNr  
 Diagnose  
 PatNr  
 GenusKonten

SELECT P.NAME PNR ~~P.DIAG~~ R.DIAGNOSE

FROM P, R

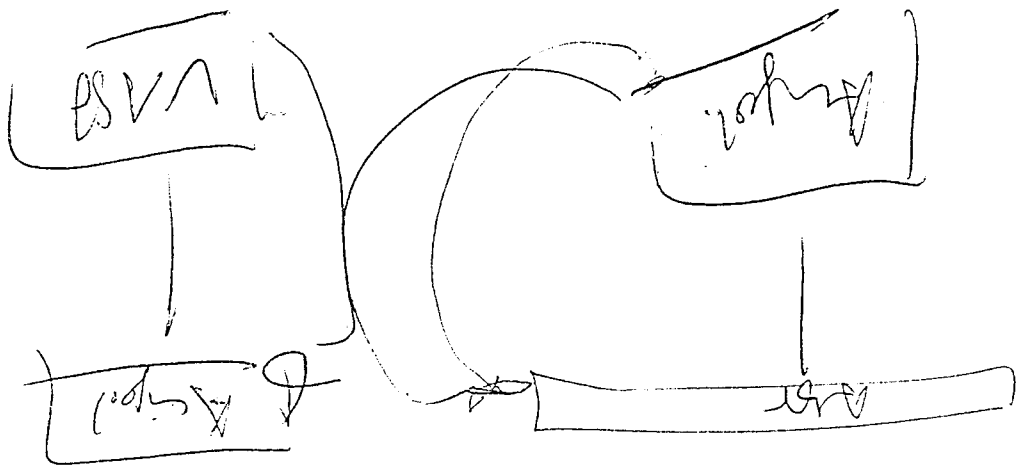
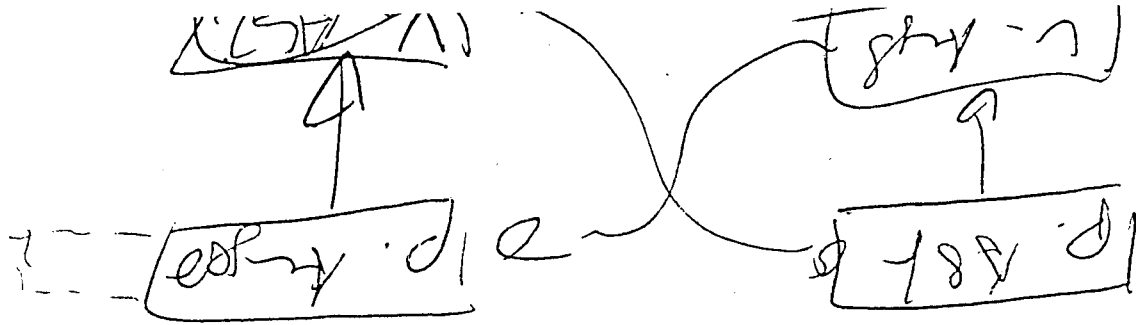
WHERE P.GEB='1930' AND P.~~NAME~~ = R.PNR

RA : ~~P~~  
 P.NAME, P.PNR, R.DIAG (P)  
 RECHNUNG  
 P.M  
 GEB='1930'  
 = O.PNR  
 (PARTENT)

Patient (P) v Diagnose (R)  
 39 Patient (P) v ~~Diagnose~~ (R) v  
 P.NAME, P.PNR, R.DIAG  
 P.GEB='1930' v P.PNR = R.PNR

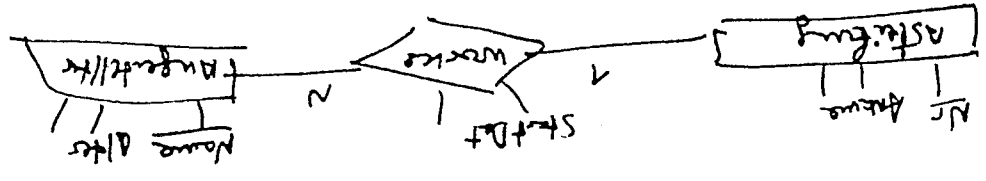
} vut | Patient (v, u, '1930') v ~~Diagnose~~  
 RECHNUNG (R, T, u, g)





SELECT ANG. NAME  
 FROM ANGESTELLE ABTEILUNG  
 WHERE A-ALTER > 54 AND AG-NAME = 'F. & E'  
 AND N-ANG = AG-ANG

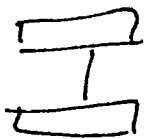
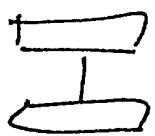
Abteilung	Anger	Anger	Anger
Anger	Anger	Anger	Anger
Anger	Anger	Anger	Anger
Anger	Anger	Anger	Anger



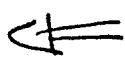
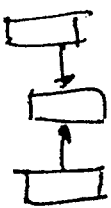
SPRACHEN  
 BEWERTUNGSMODELLE  
 THEORETISCH  
 rel. Algebra  
 rel. Kalkül  
 PRAKTISCH  
 SQL - Entw. von IBM  
 QUEL - für INGRES

NETZWERK MODEL : computer Attribute erlausst!

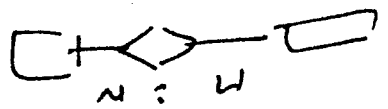
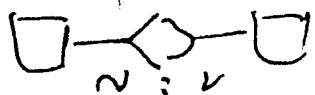
1 member R+



linking record



or



**Prüfungsprotokoll Praktische Informatik**

**Prof. Jarke**

Datum: 7.12.95

Themen: Einf. in Datenbanken

Implementierung von Datenbanken

Betriebssysteme (Silberschatz/Galvin: Operating System Concepts)

Begonnen hat Jarke mit Betriebssystemen: „Wollen wir doch mal einige Zusammenhänge zwischen Betriebssystemen und Datenbanken diskutieren. Der gleichzeitige Zugriff von Transaktionen auf Datenobjekte muß ja synchronisiert werden, welche Methoden gibts es denn dazu?“

Als Softwarelösung gibts es zum Beispiel den Bakery-Algorithmus...

Wie sieht der denn aus? Schreiben Sie mal hin!

Bakery-Algorithmus wie im Buch hingeschrieben.

OK. Welche Konzepte kennen Sie noch ?

Semaphoren. Definition von Wait und Signal hingeschrieben und die Realisierung mit einer FIFO-Warteschlange (um Busy-Waiting zu vermeiden!) erklärt.

Da gibt es ja das Consumer-Producer Problem. Erklären Sie das mal und schreiben Sie mal eine Lösung dafür mit Semaphoren hin.

Ich hab folgendes hingeschrieben:

```
Producer
Wait(S)
IF counter < n
THEN BEGIN
  add_element_to_buffer
  counter := counter + 1
END
Signal(S)
Remainder Section (Produzieren)

Consumer
Wait(S)
IF counter > 0
THEN BEGIN
  get_element_from_buffer
  counter := counter - 1
END
Signal(S)
Remainder Section (Konsumieren)
```

und erklärt, daß der counter die Füllung des begrenzten (!) Puffers angibt. Jarke war damit nicht so ganz einverstanden und sagte, daß es bei dieser Lösung noch ein Problem gäbe...

*Wahrscheinlich ist entweder Starvation oder Deadlock nicht verhindert...*

Was passiert denn, wenn der Producer ein neues Element produziert hat und dann merkt, daß der Buffer voll ist? Besser wäre doch, wenn er nur dann ein neues Element produziert, wenn auch Platz im Puffer ist...

*Ich gebe Ihm recht aber komme nicht ganz auf die richtige Idee, meine Lösung dahingehend zu verändern. Jarke meint daraufhin, ich sollte einen zweiten Semaphore einführen. Ich komme aber immer noch nicht auf die rechte Lösung. Schließlich sagt der Besitzer, daß es auch mit nur einem Semaphore geht! Ich bin inzwischen völlig durcheinander und überlasse die Diskussion Herrn Jarke und seinem Assi...*

*Das Problem sollte man sich also vor der Prüfung mal klar machen!*

Schließlich: Was das Sperren von Datenobjekten angeht, da gibt es ja das Multigranulare Sperrprotokoll. Malen Sie mal diesen Baum auf und erklären Sie das Ganze...

*Gesagt getan. (Locks werden Top-Down gesetzt und BottomUp aufgehoben)*

Schreiben Sie mal die Kompatibilitätstabelle für die Intension-Locks auf!

*Aufgeschrieben und erläutert.*

Was bedeutet *rwil* und wofür braucht man das?

*rwil = Read-Intension-Write-Lock, diesen Lock setzt man z.B. auf ein File, wenn man das ganze File lesen und vielleicht einige wenige Tupel im File ändern möchte.*

OK. Um nochmal einen Bezug zu Betriebssystemen herzustellen betrachten wir mal die 5 Schichten-Architektur. Wieso kann es sinnvoll sein, die Systempunktverwaltung vom DBMS statt vom Betriebssystem durchführen zu lassen?

*1. Sicherheitsgründe bzw. Unterstützung des Recovery. Wenn man z.B. nur REDO-Recovery vorsieht, dürfen vor dem COMMIT einer Transaktion keine DB-Seiten in die stabile DB geschrieben werden. Ferner müssen vor dem Auslagern von DB-Seiten immer zuerst die dazugehörigen Log-Einträge auf Platte geschrieben werden. Das erkennt das Betriebssystem nicht.  
2. Performance: DBMS kann andere Seitenausstrategie bevorzugen (z.B. LRU-K statt LRU) und hat auch mehr Möglichkeiten, Prefetching zu betreiben.*

## Prüfungsprotokoll : DB + BS

Prüfer : Prof. Jarke  
Termin : 12.1.96

Fächer : Einführung - Implementierung DB

Betriebssystem von Silberschatz

Anfangen hat es wohl mit dem Betriebssystem, jedoch gab er die Frage nicht getrennt  
jedenach die Thema sondern gemischt.

- Bakery-Algorithmus :  
Das Problemendarstellung, die Idee des Algorithmus erklären. Jedoch wollte er nicht  
explizite Angabe des Algo.

- Semaphore :  
Explizit aufschreiben und erklären, mit Busywaiting und die Vermeidung

- Deadlock :  
Die Bedingungen und die Vermeidung, Verhindern, Erkennen und beseitigen.  
Eine Vorlesung halten. Er stellt einige kleine Frage dabei.

- In DB-System :  
Wie wird Deadlock in DB-System verhindert? In anderen Protokolle und Trans-  
aktionsverwaltung schauen.

- Short-term Scheduler :  
Methode aufzählen und besonders mit dem Bild (siehe im Buch von Silberschatz)  
erklären, wie die Vorgang ist.

- 2PL-Scheduler :  
Was es ist, welche Probleme da gibt's (Deadlock), Bild malen, in welchen Phase  
aufritt, ein Bsp von Schedule mit Deadlock angeben, die Lösungsweg.... Hier  
muss man viel wissen.

- B\*-Baum :  
Wie sieht er aus, die Kosten, ....wie übliche

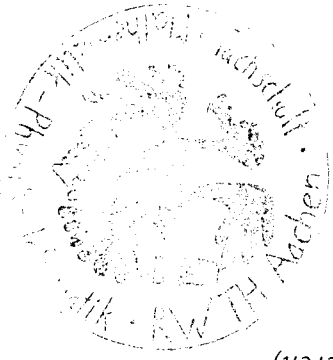
- relationale Algebra :  
alle sorte von Operatoren angeben

- Dekomposition :  
zu beachten : Verlustlosigkeit, Abhängigkeitserhaltung. Die genaue Definition und  
erklärung  
Die Test-Methode.

Viele sagen, dass der Professor sein Stil geändert hat. Aber bleibt die Frage immer in  
selben Thema, finde ich. Ich habe kein einzige Frage gehabt, die nicht in Prüfungs-  
protokoll stand. Nur muss man ein bisschen tiefer kennen. viel glück.

Sehen ersetzungsstrategie  
'Pre Be tclung', .....

Prof. Jarke  
m. Sc. Zeilermann  
Lösung BS



```

Producer
Wait(S)
IF counter < n
THEN BEGIN
  add_element_to_buffer
  counter := counter + 1
END
Signal(S)
Remainder Section (Produzieren)

Consumer
Wait(S)
IF counter > 0
THEN BEGIN
  get_element_from_buffer
  counter := counter - 1
END
Signal(S)
Remainder Section (Konsumieren)

```

Ich hab folgendes hingeschrieben:

Da gibt es ja das Consumer-Producer Problem. Erklären Sie das mal und schreiben Sie mal eine Lösung dafür mit Semaphoren hin.

Semaphoren. Definition von Wait und Signal hingeschrieben und die Realisierung mit einer FIFO-Warteschlange (um Busy-Waiting zu vermeiden!) erklärt.

OK. Welche Konzepte kennen Sie noch ?

Bakery-Algorithmus wie im Buch hingeschrieben.

Wie sieht der denn aus? Schreiben Sie mal hin!

Als Softwarelösung gibts es zum Beispiel den Bakery-Algorithmus...

Begonnen hat Jarke mit Betriebssystemen: „Wollen wir doch mal einige Zusammenhänge zwischen Betriebssystemen und Datenbanken diskutieren. Der gleichzeitige Zugriff von Transaktionen auf Datenobjekte muß ja synchronisiert werden, welche Methoden gibts es denn dazu?“

<b>Prüfungsprotokoll Praktische Informatik</b>	
<b>Prof. Jarke</b>	
Datum:	7.12.95
Themen:	Einf. in Datenbanken Implementierung von Datenbanken Betriebssysteme (Silberschatz/Galvin: Operating System Concepts)

und erklärt, daß der counter die Füllung des begrenzten (!) Puffers angibt. Jarke war damit nicht so ganz einverstanden und sagte, daß es bei dieser Lösung noch ein Problem gäbe...

*Wahrscheinlich ist entweder Starvation oder Deadlock nicht verhindert...*

Was passiert denn, wenn der Producer ein neues Element produziert hat und dann merkt, daß der Buffer voll ist? Besser wäre doch, wenn er nur dann ein neues Element produziert, wenn auch Platz im Puffer ist...

*Ich gebe Ihnen recht aber komme nicht ganz auf die richtige Idee, meine Lösung dahingehend zu verändern. Jarke meint daraufhin, ich sollte einen zweiten Semaphore einführen. Ich komme aber immer noch nicht auf die rechte Lösung. Schließlich sagt der Besitzer, daß es auch mit nur einem Semaphore geht! Ich bin inzwischen völlig durcheinander und überlasse die Diskussion Herrn Jarke und seinem Assi...*

Schließlich: Was das Sperren von Datenobjekten angeht, da gibt es ja das Multi-gramulare Sperrprotokoll. Malen Sie mal diesen Baum auf und erklären Sie das Ganze...

*Gesagt getan. (Locks werden Top-Down gesetzt und BottomUp aufgehoben)*

Schreiben Sie mal die Kompatibilitätstabelle für die Intension-Locks auf!

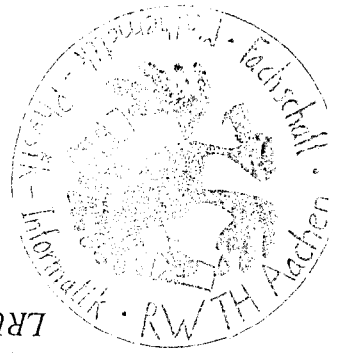
*Aufgeschrieben und erläutert.*

Was bedeutet rwl und wofür braucht man das?

*rwl = Read-Intension-Write-Lock, diesen Lock setzt man z.B. auf ein File, wenn man das ganze File lesen und vielleicht einige wenige Tupel im File ändern möchte.*

OK. Um nochmal einen Bezug zu Betriebssystemen herzustellen betrachten wir mal die 5 Schichten-Architektur. Wieso kann es sinnvoll sein, die Systempufferverwaltung vom DBMS statt vom Betriebssystem durchführen zu lassen?

1. *Sicherheitsgründe bzw. Unterstützung des Recovery. Wenn man z.B. nur REDO-Recovery vorsieht, dürfen vor dem COMMIT einer Transaktion keine DB-Seiten in die stabile DB geschrieben werden. Ferner müssen vor dem Auslagern von DB-Seiten immer zuerst die dazugehörigen Log-Einträge auf Platte geschrieben werden. Das erkennt das Betriebssystem nicht.*  
2. *Performance: DBMS kann andere Seitenausstrategie bevorzugen (z.B. LRU-K statt LRU) und hat auch mehr Möglichkeiten, Prefetching zu betreiben.*





Erklären Sie bitte, was Monitore sind, am besten an einem konkreten Beispiel. Also erkläre, daß es ein Konstrukt ist, mit dem Prozesse synchronisiert werden, die auf gemeinsame Datenstrukturen zugreifen wollen. Im Monitor darf sich immer nur ein Prozess zur gleichen Zeit aufhalten, womit Mutual exclusion garantiert ist. Ein Beispiel wäre das Producer-Consumer-Problem, wobei ein Monitor dazu genutzt werden könnte, den Zugriff auf den gemeinsamen Buffer zu regeln. (Er wollte wohl noch, daß ich dieses Beispiel konkret als Monitor hinschreibe, aber irgendwie habe ich es geschafft, mich darum zu drücken...)

Dies ist ja nun ein Synchronisationsmechanismus für zwei Prozesse. Kennen Sie noch einen für mehrere Prozesse? Den Bakery-Algorithmus. Sollte ihn dann Punkt für Punkt aufschreiben und erklären.

Können dort Deadlocks auftreten? Nein, denn es geht immer nur um den kritischen Bereich, also um ein Betriebsmittel.

Wenn man so schöne Softwarekonstrukte für die Prozesssynchronisation hat, warum braucht man dann überhaupt noch Hardwarekonstrukte wie die Semaphoren? Um busy waiting zu vermeiden. Bei den Semaphoren gibt es die Möglichkeit, wartende Prozesse in eine Warteschlange einzureihen.

Bei mehreren Prozessen können ja nun Deadlocks auftreten. Nennen Sie doch mal die vier Bedingungen für Deadlocks.

Mutual exclusion, no preemption, hold and wait, circular wait.

## Expertensysteme

Das sollte reichen zu Betriebssysteme.

Wenn Sie nun einmal Datenbanksprachen wie SQL oder Relationenkalkül und EFRS vergleichen, was können Sie zu deren Mächtigkeit sagen?

SQL und Relationenkalkül sind insofern mächtiger, als daß sie auch Negation und Disjunktion ausdrücken können. Auf der anderen Seite ist EFRS angenehmer, weil entscheidbar ist, ob eine Formel allgemeingültig ist. Beim Relationenkalkül, das ja auf der Prädikationlogik erster Stufe basiert, ist dies nicht der Fall.

(sieht mich etwas verdutzt an) Was soll denn das bedeuten (kannte er wohl wirklich nicht)? Das bedeutet, daß für EFRS ein Algorithmus existiert, der entscheiden kann, ob eine vorliegende Regel allgemeingültig ist.

Achso. Na gut, aber was können Sie denn mit EFRS erreichen, was bei den relationalen Sprachen unmöglich ist?

Man kann neue Fakten ableiten aus den bestehenden Regeln und Fakten. Bei Datenbanken kann man nur bereits existierende Fakten abfragen.

Naja, aber bei Datenbanken erhält man ja auch neue Informationen, wenn man z. B. einen Join macht. Da gibt es aber noch etwas anderes...

(Hm, was will er denn nur hören? Ein letzter Versuch.)

Man kann bei EFRS garantieren, daß alle Anfragen endlich sind, da Cons(S) endlich ist.

# Gedächtnisprotokoll Diplomprüfung in Praktischer Informatik

bei: Prof. Jarke

am: 21. April 1993

Dauer: exakt 3/4 Std.

Fächer: Einführung in Datenbanken

Implementierung von Datenbanken (Vorlesung Kemper)

Betriebssysteme (Buch Silberschatz ...)

Fragen:

Erklärung des Entity-Relationship-Modells.

Beispiel aufmalen für:

Datei Angestellte mit Name und Adresse;

Datei Abteilung mit Nummer und Titel

und Beziehung gehört-zu mit Einstellungsdatum

Jeder Angestellte gehört eindeutig zu einer Abteilung

Umsetzung dieses Beispiels in das relationale Datenbankmodell.

Wie kann man dies noch vereinfachen?

Relationen mit gleichen Schlüsseln zusammenfassen.

Jetzt betrachten wir das Modell mal als n:m - Beziehung.

Welche Probleme treten auf?

Anomalien, Redundanzen

Welche Normalform haben wir hier vorliegen?

Warum? Welche Abhängigkeit stört?

Was kann man nun dagegen machen?

Was ist Dekomposition?

Was soll denn bei der Normalisierung erhalten bleiben?

Erklärung von verlustlos und abhängigkeitserhaltend.

Wie kann man dies beweisen?

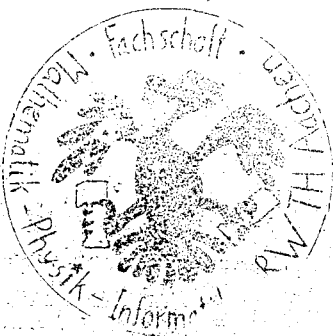
Tableaumethode für Verlustlosigkeit (mit Erklärung)

Abschlussberechnung für Abhängigkeitserhaltung

Wie berechnet man den Abschluss?

mit den Armstrong Axiomen

Synthese-Algorithmus kurz erklären.



PKL

- Folgende Anfrage in SQL hinschreiben:
- Alle Angestellten, die in Aachen wohnen und in der Forschung arbeiten.
- Welche Arten von Anfragesprachen gibt es?
- Welche SQL-Statements entsprechen welchen algebraischen Operatoren?
- Die Anfrage in relationaler Algebra hinschreiben.
- Wie kann man diese Anfrage optimieren?
- Überleitung zur Implementierung:
- Wie wird der Join implementiert?
- Welche Probleme treten bei parallel laufenden Transaktionen auf?
- Was ist *serialisierbar*?
- gleiches Ergebnis wie bei serieller Abfolge.
- Womit erreicht man das?
- Sperren setzen und 2-Phasen-Sperreprotokoll.
- Wieso werden Probleme dadurch gelöst?
- Was ist wenn nach der Freigabe von Sperren wieder Sperren angefordert werden können?
- Welches sind die vier Bedingungen für einen Deadlock? Erläuterung.
- Was kann man beim 2-Phasen-Sperreprotokoll tun, um Deadlocks zu vermeiden?
- Bei welchen Speicherungsstrukturen wird das gemacht?
- B-Bäume
- Wie werden Sperren gesetzt, damit nicht der gesamte Baum blockiert wird?
- Noch was zu Betriebssysteme:
- Was ist Virtual Memory?
- Wie wird es realisiert?
- Demand Paging
- Welche Seitensetzungsstrategien gibt es? Erklärung.
- Kennen Sie ein Betriebssystem, wo ein bestimmtes (welches?) angewandt wird?
- In welche Schicht des 5-Schichtenmodells gehört das?
- Systempufferschnittstelle.



Gedächtnisprotokoll  
Diplomprüfung Vertiefungsgebiet Informationssysteme  
Professor Jarke  
Dezember 1996  
Dauer: 60 Minuten

Prüfungsvorlesungen:

Einführung DB (Jarke)

Implementierung DB (Jarke)

Verteilte DB (mie gehört, aber Mitschrift der Vorlesung von Jeusfeld)

Requirements Engineering (Pohl)

### 1) Requirements Engineering (ca. 25 Minuten)

F: Hilfsmittel beim RE?

A: SA: DFDs, Minispezifikationen und DS

F: Was steht in den DS

A: Daten über Daten, d.h. Metadaten

F: DFD-Symbole erklären

A: 1) Prozesse/Knoten als Kreise darstellen

2) Datenspeicher als parallele Linien

3) Datenflüsse als gerichtete Kanten

4) Datenquellen oder -senken als Rechtecke

F: Welche „Constraints“ gibt es in DFDs, d.h. was sollte nicht entworfen werden ?

A: 1) Datenspeicher, der nur Daten aufnimmt, oder nur Daten abgibt (tritt relativ selten auf, kann es aber durchaus geben)

2) Datenflüsse/Kanten immer benennen

3) Prozesse: es muß i.a. Kanten geben, die herein laufen und welche, die herausgehen

F: Was macht man vor der Erstellung des DFD ?

A: Man geht zum Kunden und erstellt mit ihm ein Konzept

(Jetzt wurde es etwas unangenehm)

F: Und was macht man, wenn das nicht geht ?

A: Man analysiert das derzeit vorhandene System

F: und wenn es das nicht gibt ?

A: (Der Gedanke: „dann entwickelt man einfach eines und stellt es ihm vor“, war mir zu einfach)

Ich fing an etwas über die 4 Welten des RE (unterschiedlich Ansichten von einem System) und die 3 Dimensionen des RE-Prozesses (Spezifikation, Repräsentation und Agreement) zu erzählen.

(Was er wirklich wissen wollte, blieb mir verborgen - vielleicht wollte er nur einen allgemeinen Begriff wie „Brain-Storming“ hören ?)

F: Wie sollte ein gutes DFD aussehen ?

A: nicht zu groß (ca. 6-7 Prozesse), sonst aufgliedern in mehrere Ebenen und Projektion eines Knotens (welcher ein Teilsystem darstellt) auf eine Ebene

F: Was ist denn hier unbedingt (constraints) zu beachten ?

A: (nach mehrmaligem hin und her) ... Sie meinen bestimmt, daß die mit der Umgebung in Verbindung stehenden Kanten die gleichen sein müssen (trivial)

möglichst viele Prüfungsprotokolle (ca. 30-50)

Buch Silberschatz/Korth

**Empfehlung:** Buch Vossen

**Empfehlung:** Ich kann die Prüfung in DB / Impl. DB nicht unbedingt weiter empfehlen. In den Vorlesungen zu viele Themen nur angeschnitten und die Folienkopien sind schwer verständlich. Die Literatur (Vossen) umfaßt ca. 500 Seiten, deckt aber nur 80 % des Stoffes der Vorlesungen DB und Impl. DB ab. Seht Euch auf jedem Fall die Prüfungsprotokolle an, so könnt Ihr leichter abschätzen, was Ihr unbedingt lernen solltet und was nicht. Sammelt alles über das Schichtenmodell und versucht soviel zu verstehen, daß Ihr über jede einzelne Schnittstelle ein paar Minuten referieren könnt.

**Fazit:** Note 2,7 - ich hatte mehr erwartet. Angenehm war, daß er nichts zu Quantraphen Einfügen/Löschen in B/B\*-Bäumen, o.ä. insbesondere aus Kapitel 2 Implementierung fragte.

**F:** einige Fragen, wie z.B. "Wie implementiert man das XXX zwischen Schicht i und i+1"

**A:** 6 Schnittstellen aufgeschrieben und angefangen zu reden über alles, was mir dazu einfällt ... dabei B/B\* - Bäume erwähnt, Hash erklärt, Nested-Loop und Merge/Sort-Join, Komplexität des Nested-Loop =  $n^2$ , des Merge/Sort-Join  $n \cdot \log n$  (geschätzt und er hat genickt) ...

**F:** Na gut, wenn Sie das nicht können ... Kommen wir zur Impl. DB (es waren schon ca. 45-50 Minuten vergangen) 6 Schnittstellen der 5 Schichtenarchitektur und eine SQL-Anfrage sind gegeben. Erklären sie die Schnittstellen mit Hilfe der SQL-Anfrage:

**3) Implementierung von Datenbanken (10-15 Minuten)**

**A:** Habe gefragt, ob es wirklich ganz formell haben möchte, denn dazu müßte ich zuerst mal die Mengen (Attributmengen) festlegen ?

**F:** Genauer ?

**A:** Verbal erklärt (A->B : wenn 2 Tupel auf A übereinstimmen, dann auch auf B)

**F:** Wie ist eine FD definiert ?

**A:** Dekompositions- und Syntheselgorithmen hat er nicht abgefragt ! (Dekompositionen und verlustfrei). Verlustfreiheit erklärt. Abhängigkeitserhaltung erklärt.

**A:** Überführt in BCNF, verlustfrei aber nicht immer abhängigkeitserhaltend. Synthese überführt in 3. NF (abhängigkeitserhaltend und verlustfrei).

**F:** Was macht die Dekomposition ?

**A:** select nr,count(nr) from relation group by nr having (count(nr)>1) -- oder ähnlich --

**F:** Sie können das erste select weglassen, wenn Sie noch ein having anfügen

überhaupt nichtig sei)

**A:** select nr from relation where count(nr)>1 and nr in (select nr,count(nr) from relation group by nr) ... (oder so ähnlich, aber dann gemeint, daß das wohl zu komplex würde, abgesehen davon, ob es

der Datenbank vorkommen.

**F:** Machen Sie doch mal eine SQL-Anfrage: Sie suchen alle Buchnummern, die mehr als 1x in (zwischen durch meinte er mal, daß wir nicht schnell genug voran kämen).

Möglichkeiten, die offensichtlich alle OK waren)

**A:** Entitäten -> Relationen; mehrwertige Attribute -> Relationen; Beziehungen -> Relationen (oder bei 1:1 oder 1:n die Attribute an die Relationen der Entitäten hängen); ISAs -> (mehrere

**F:** Wie überführt man so etwas in Relationen ?

**A:** Kardinalitäten aufgezählt (er wollte die 0:1 1:1 0:n 1:n m:n - Notation haben !)

**F:** Constrains ?

**A:** Entitäten, Beziehungen, Attribute (auch mehrwertige), Schlüssel, ISA-Bez.

im ER-Diagramm ?

**F:** Genau, aber so trivial ist das gar nicht ... Gehen wir über zu DB. Welche Objekte gibt es

**2) Einführung in Datenbanken (20-25 Minuten)**

Und aus der neueren Auflage des Elmasri/Navathe das Kapitel über deduktive DB. Für mich war bei meiner Fächerkombination völlig selbstverständlich, daß die Prüfung mit Einführung von DB beginnen müsse, deshalb überraschte mich Jarke's Frage:

*Mit welchem Thema sollen wir beginnen?*

Mit Einführung von DB.

*Haben Sie Implementierung bei mir oder bei Kemper gehört?*

Auch diese Frage ließ mich zusammensucken. War es wohl ein Nachteil Jarke's Vorlesung angegeben zu haben, während wohl die meisten Prüflinge Kemper hatten. Oje, mir schwante Böses. (Aber so schlimm wurde es dann bei Implementierung doch nicht.)

**Einführung von DB: (20 min)**

*Ja, dann fangen wir doch mal recht standardmäßig mit Einführung an. Wie geht man denn beim DB-Entwurf vor?*

(Naja, der Standard blieb mir leider weitgehend verborgen.) Man stellt ER-Modell auf, indem man Entitäten, Beziehungen, Rollen und Attribute bestimmt. Dann leitet man daraus Relationenschemata ab und normalisiert diese bei Bedarf.

*Nein, nein, worauf ich hinaus will, was macht man vor der ER-Modellierung?*

(Tja, was macht man da, dieser Teil der Prüfung war grausam, das 5. Kap. (aber auch das 4. Kap.) aus dem Einführungsskript sollte hier wohl abgeprüft werden, aber das konnte ich leider nicht, hab mir also mit gesundem Menschenverstand versucht, etwas zu überlegen. War aber im Ergebnis nur betriebliegend, also glaub meinen Antworten eher nicht.) Tja, der DB-Entwickler versucht mit dem Kunden (Benutzer, Administrator) Interviews oder Brainstormings zu führen, um die notwendigen Informationen zu beschaffen / die Begriffe und Objekte zu klären, die wesentlich für die Entwicklung der DB sind. Diese werden dann zu Entitäten etc. im ER-Modell. Darüber hinaus Beziehungen und Abhängigkeiten der Objekte untereinander ergütenden, um daraus funktionale Abhängigkeiten ableiten zu können.

*Welche Verfahren gibt es, um aus gegebenen Attributen und funktionalen Abhängigkeiten Relationenschemata abzuleiten?*

Dekompositions- und Syntheseverfahren. Bei der Dekomposition bestimmt man transitiven Abschluß der funktionalen Abhängigkeiten (FD) und wählt dazu minimale Überdeckung. Bei der Synthese bildet man Äquivalenzklassen der FDs und wählt einen der äquivalenten Schlüssel mithilfe der charakteristischen Menge. (Hier wollte Jarke es mal wieder ziemlich genau wissen und fand meine Ausführungen nicht exakt genug. Er meinte, ja in den Lehrbüchern steht das wohl nicht so genau drin.)

*Was können die beiden Verfahren jeweils garantieren?*

Dekomposition die Verlustfreiheit, Synthese die Abhängigkeitserhaltung. (Begriffe jeweils erläutert.)

*Wie kann man die Einhaltung der Verlustfreiheit überprüfen?*

(weiß nicht)

*Sagt Ihnen Tableaumethode etwas?*

Ja, also da trägt man in die Tabelle aller beteiligten Attribute für jede Teilrelation eine Zeile ein. In jeder Zeile vermerkt man, welche Attribute durch die Teilrelation überdeckt werden. Ziel ist es, eine total überdeckte Zeile zu bekommen. Aber wie man daran kommt, weiß

Über die Verteilungen der vorhandenen Attributwerte, d.h. über die Anzahl verschiedener Attributwerte bei Schlüsselattributen, über die der Join gebildet wird.

Die konkreten Anzahlen weiß man aber nur, wenn man die Relationstempel zunächst alle betrachtet. Das wäre zu teuer. Was nimmt man vereinfachend an?

Man nimmt allgemeine, zu erwartende Verteilungen der Werte an, wobei man oft eine Gleichverteilung voraussetzt.

*Ja, obwohl auch da Kritiker nicht voll zufrieden sind. Warum?*

Unrealistisch. Eine Umsortierung von (mehr als 2) beteiligten Join-Operationen erfolgt auf der Annahme der Gleichverteilung. Liegt diese Gleichverteilung im konkreten Fall jedoch nicht vor, werden die Zwischenergebnisrelationen entscheidend größer als erwartet. Die Umsortierung kann dann evtl. sogar teurer ausfallen als eine, die theoretisch schlechter ist.

*Kommen wir nun zu Betriebssystemen.*

(Panik in den Augen, denn es verbleiben noch 20 min. Doch nicht jetzt schon, da hab ich doch keine Ahnung.)

*(sieht wohl mein Entsetzen) Oder möchten Sie lieber, daß ich Sie etwas über Transaktionen frage?*

Ja bitte, Transaktionen. (Nun begann meine Sternstunde, Jarke ließ mich größtenteils gewähren und über das Thema frei referieren. Er stellte nur ab und an Zwischenfragen. Ob er nun die folgenden Fragen in der Form gestellt hat, weiß ich nicht mehr. Ich weiß nur, daß ich das, was in den Antworten steht, erzählt habe und zwar im wesentlichen zu dem Bild des 2-Phasen-Sperreprotokolls.)

*Welches Prinzip muß bei Transaktionen gewahrt werden?*

ACID-Prinzip. Atomarität (vollständig oder gar nicht), Konsistenz (DB vor und nach der Transaktion in konsistentem Zustand), Isolation (DB theoretisch für jede Transaktion exklusiv, da Daten gesichert), Persistenz/durability (wenn Transaktion erfolgreich endet, sind durch sie erzeugte Effekte persistent geworden)

*Welche Synchronisationsprobleme gibt es?*

Lost-Update-, Dirty-Read-, Phantom-Problem.

*2-Phasen-Sperreprotokoll (2PhaseLocking, 2PL).*

Wachstumsphase, in der die Sperren aller Transaktionen gesetzt werden; Phase, in der die Transaktionen arbeiten; "Schrumpf"phase, in der die Sperren wieder freigegeben werden. In der Schrumpfphase können kaskadierende Aborts (näher erläutert) auftreten ==> um das zu vermeiden, am besten steile Flanke (strenges 2PL).

Problem, wann bekommen Transaktionen "commit", d.h. wann werden sie als erfolgreich bezeichnet und wann werden ihre Aktionen persistent. Wenn commit zu früh kommt und beim Kaskadieren eine andere Transaktion fehlschlägt, muß commit-Transaktion zurückgesetzt werden (UNDO). Wenn commit später kommt, kann in der Zwischenzeit ein Systemfehler aufgetreten sein, wodurch Commit-Transaktion noch nicht persistent gemacht werden konnte (noch nicht in der stabilen DB). Dann wird ein REDO dieser erfolgreicheren Transaktion erforderlich, die dazu notwendigen Informationen stehen im LOG-File. Recovery-Protokolle können beliebige Kombinationen von UNDO und REDO enthalten (weder UN- noch REDO, nur UNDO, nur REDO, nur UNDO, sowohl UN- als auch REDO).

(Die Deadlock-Problematik in der Wachstumsphase hatte ich absichtlich nicht von mir

Fach: Praktische Informatik

Vorlesungen: Einführung in Datenbanken (Jarke WS92/93)

Implementierung von Datenbanken (Jarke WS93/94)

Betriebssysteme (Buch: Silberschatz Kap. 1-12 + Kap. 19 (Unix))

Prüfer: Prof. Jarke

Dauer: 45 Minuten

Zeit: 1. Quartal 95

Note: 1,3

### Einführung in Datenbanken

Wir haben uns in der Vorlesung hauptsächlich mit dem relationalen Modell befaßt. Ein Datenmodell zeichnet sich durch Strukturen, Operationen und Integritätsbedingungen aus. Erklären Sie diese bitte!

Relationen: kartesisches Produkt der Wertemengen der Attribute

Tupel: Elemente von diesen

Operationen: Algebra/Kalkül, Operationen der Relationenalgebra aufgeschrieben

(zu den Integritätsbedingungen bin ich nicht mehr gekommen)

Gibt es bei den Algebraoperationen irgendwelche Einschränkungen?

$\cap$ ,  $\cup$ ,  $-$ : verträgliche Attributmengen

Nun zu praktischeren Anfragesprachen: SQL. Wie übersetzt man einen SQL-Ausdruck in

Algebra?

S-F-W-Block: kartesisches Produkt, Selektion, Projektion

Drücken Sie bitte die Operation  $\cap$  in SQL aus!

(von da an kam ich ein wenig ins Schleiudern, obwohl ich dachte, gerade diese

Anfrageformulierungen gut zu beherrschen: daher meine Empfehlung: bimsen!)

SELECT A1, ..., An

FROM R1, R2

WHERE R1.A1 = R2.B1 AND ...

Und jetzt die Mengendifferenz!

Meine Lösungen waren irgendwie ziemlich skurril. Eine schriftliche Weiterverbreitung

waren sie auf keinen Fall wert.

Gegeben ist folgendes Schema:

EMPL(eno, ..., sal, dno)

DEP(dno, dname).

Formulieren Sie bitte eine Anfrage, welche die Namen (dname) von Departments liefert, in denen alle Angestellten mehr als 50.000 verdienen!

Nun, hier hatte ich irgendwie einen Blackout und mußte einigermaßen rumprobieren und habe es dann doch nicht richtig hingekriegt. Für die Note war das scheinbar nicht

Prüfungsprotokoll vom: 20. 1. 1994  
 Prüfer: Jarke/Nejdl  
 Fächer: Einführung DB (Jarke-Skript, Ullmann, etc.),  
 Expertensysteme (Gottlob/Frühwirth),  
 Wissensrepräsentation (Genesereth/Nilsson)  
 Dauer: ca. 50-55 Minuten  
 Note: 1.7

Meine Prüfung war insofern etwas ungewöhnlich, als Jarke's erste Frage lautete, ob ich etwas dagegen hätte, wenn nicht er sondern Nejdl mir die Fragen zur Wissensrepräsentation und zu Expertensystemen stellen würde. Mir war's prinzipiell egal, und Nejdl wusste natürlich besser, was in der Vorlesung tatsächlich dran war, also war ich einverstanden (im Nachhinein wahrscheinlich eine gute Idee).

## 1 Datenbanken

Jarke: beschreibt Kunde-Waren-Einkauf-Zusammenhänge, möchte ER-Diagramm.  
 ich: male selbiges hin  
 Jarke: läßt Relationen aufstellen  
 ich: Relationen Kunde: (Kundennr), Ware: (Warennr, Preis, Gruppe) und Einkauf: (KNR, WNR, Menge)  
 Jarke: beschreibt Supermarkt-Szenario und will SQL-Anfrage, die die Anzahl der gekauften Produkte mit Warengruppe für einen Kunden auflistet  
 ich: (kriege die Anfrage mit Müh und Not hin)  
 Jarke: weitere SQL-Anfrage: Aufsummieren der Gesamtmenüen, die von jedem Produkt gekauft wurden  
 ich: (habe mit SQL so genau nicht angeguckt, kann mich daher nur noch an die sum-Funktion erinnern, auf das *group by* bin ich nicht mehr gekommen)  
 Jarke: malt mir ein Rechnungsformular hin und möchte die entsprechenden Relationen in 1NF und 3NF (dabei soll ich die Normalformen erklären)  
 ich: erkläre NF, male 3NF hin (er will aber erst die 1NF, na gut). Leider macht das wohl keinen so souveränen Eindruck - er hat mich aber auch ziemlich durcheinander gebracht mit seiner SQL und diesem Beispielformular (das übrigens genau so im Skript steht, und das ich immer überschlagen hatte).  
 Jarke: Danke, das war's zu Datenbanken  
 ich: bin etwas platt, normalerweise fragt er da VIEL mehr zu - und darauf wäre ich auch vorbereitet gewesen

## 2 Expertensysteme

Nejdl: Welche Formalismen wurden zur Wissensrepräsentation vorgestellt?  
 ich: PLI und Untermenü davon, EFRS  
 Nejdl: Was ist der Unterschied?

- ich: (denk) EFRS haben keine Negation, keine Funktionensymbole (da war noch mehr, bloss was?)
- Nejdl: Was kann man in EFRS nicht? ich: (überlege krampfhaft, mit einigen Tips von Nejdl) Disjunktion von pos. Literalen in einer Klausel nicht darstellbar, nur Hornklauseln möglich (?) Nejdl: Wie ist in Prolog die Negation realisiert?
- ich: erkläre *negation as failure*
- Nejdl: will *Frames* erklärt haben
- ich: male ihm einen hin, erzähle was von AKO und Vererbung, Möglichkeit, Vererbung zu blockieren
- Nedjl: will Mehrfachvererbung erklärt haben
- ich: male einen *Frame eierlegendes Tier* und einen *Säugetier*, darunter ein Schnabeltier: Eigenschaften beider *Frames* werden auf das AKO-*Frame* vererbt
- Nejdl: fragt, was bei widersprüchlichen vererbten Eigenschaften (wie im Beispiel denkbar) passiert
- ich: (verteidige erstmal die Existenz eines solchen *Tiers*;) man muß sich überlegen, welche Eigenschaften man denn nun haben will (er sagt was von Strategie zur Auswahl der entsprechenden Eigenschaften, na ja, das meinte ich doch)
- Nejdl: Was ist ein *Meta-Interpreter*?
- ich: erkläre das
- Nejdl: Wie sieht der aus?
- ich: (bin langsam beeindruckt, wie gut sowohl Nejdl als auch Jarke genau die Themengebiete herausfinden, die ich nicht so gut drauf habe) male erstmal den Aufruf und die Regeln für Konjunktion, Disjunktion und Negation hin
- Nejdl: und was noch?
- ich: \*seutz\* male noch die fehlenden hin, allerdings mit etwas Nachdenken.
- Nejdl: Was für Inferenzstrategien gibt es? ich: (höre Inferenz, bin momentan auf dem völlig falschen Dampfer, weil ich an *Inferenznetze* denke) ???
- Nejdl: (legt mir das Inhaltsverzeichnis seines Buches hin) Haben sie da schon mal reingeguckt?
- ich: (werde langsam sauer) Ja, hab ich. (Später ist mir aufgefallen, daß die Frage ihre Berechtigung hatte — die EFRS, mit denen ich auch etwas Probleme hatte, kamen kaum auf den Vorlesungsfolien vor, waren aber ziemlich ausführlich im Buch beschrieben.)
- Nejdl: Also, dann beschreiben Sie mal die Verfahren.
- ich: beschreibe Vorwärtsverkettung
- Nejdl: Ist die Menge *Cons(S)* beim EFRS immer endlich?
- ich: Ja, da nur endlich viele Ausgangsvoraussetzungen existieren und keine Rekursion (er guckt komisch), ich meine, Funktionensymbole treten nicht auf und können daher auch nicht geschachtelt werden, was zu unendlichen *Cons-Mengen* führen würde.
- Ich erkläre auch noch die Rückwärtsverkettung (Stichwort Prolog), Resolution.
- Nejdl: (zeigt auf das Beispiel, das mir bei diesen EFRS Probleme gemacht hatte)
- Resolvieren Sie das mal - genau wie Prolog das macht.
- ich: erwähne beim Resolvieren, daß Prolog Tiefensuche durchführt und daß man

darmit u.U. Lösungen verliert

Nejdl: Na, warum macht Prolog dann Tiefensuche ?

ich: (nach 1 Sek. Pause) Weil sich sonst exponentieller Speicheraufwand ergibt.

Außerdem erzähle ich ihm noch, daß man bei umfangreicheren Resolutionen gut daran tut, diese durch Strategien (z.B. Stützmengenreolution) zu unterstützen (das kann nicht in der Vorlesung vor, steht aber u.a. im Genesereth-Nilsson).

Nejdl: Wann nimmt man Vorwärts-, wann Rückwärtsverkettung ?

ich: Kommt darauf an, wohn man eine größere Verzweigung des Suchraums erwartet; außerdem hängt das vom Umfang der Datenbasis und des Zieles ab. Es wäre ziemlich unsinnig, alle Konsequenzen aus einer Riesens-Datenbasis abzuleiten, wenn man bloss eine Klausel beweisen will.

Nejdl: O.K. (nimmt Genesereth-Nilsson)

### 3 Wissensrepräsentation

Nejdl: Was ist Monotonie, was ist Nicht-Monotonie ?

ich: erkläre das

Nejdl: Welche Ansätze gibt es zur Nicht-Monotonie ?

ich: CWA, PC, Circumscription, Default-Theorien, erkläre ihm letztere, bevor er Luft holen kann, mit Beispiel

Nejdl: Was ist die Grundidee bei der CWA ?

ich: erkläre anhand eines Beispiels, weise auf Anwendung in DB hin (geographische DB mit Nachbärändern)

Nejdl: Was ist Prädikatenvervollständigung ? Malt Formel hin:  $Vogel(x) \Rightarrow Fliegt(x)$ .  
Machen Sie mal.

ich: mal ihm leider versehentlich die Vereinigung von Datenbasis und Vervollständigungsformel hin (also die Äquivalenz statt der einen Richtung); er merkt das auch an, das sollte

aber nix machen. Nebenbei hatte ich ihm noch erklärt, daß die PC nur anwendbar ist, weil die Formel solitär im entsprechenden Prädikat ist, bzw. ihn darauf hingewiesen, daß nur das eine Prädikat seiner Klausel vervollständigt werden kann, weil

das andere eben nicht solitär ist.

Nejdl: Will wissen, was der Unterschied zwischen PC und CIRC ist

ich: CIRC geht auch mit anderen Klauseln, denke an Beispiel  $EyP(Y)$ , besinne mich aber, weil mir dazu keine CIRC einfällt.

Nejdl: Ja, wann geht die CIRC denn ?

ich: CIRC geht immer !

Nejdl: (stellt eine Frage, die ich akustisch nicht verstehe)

ich: Bitte ?

Nejdl: (muß noch zweimal wiederholen, bis ich das letzte Wort seiner Frage als 'Theorie' identifizieren kann.)

ich: Also, wenn Sie 'ne konsistente Theorie reinstecken, kommt auch 'ne konsistente Theorie raus; bei 'ner inkonsistenten kriegen sie auch 'ne inkonsistente, aber an-

wenden läßt sich das Verfahren immer.

Nejdl: will wissen, was denn im Spezialfall rauskommt bei einer CIRC

ich: Ach so, oftmals kollabiert die CIRC zwar zu einer Formel erster Stufe, aber

eigentlich ergibt sich eine zweiter Stufe.

Jarke: sagt, daß das jetzt doch schon reicht, oder ?

Nejdl: (will unbedingt noch eine Frage stellen, mittlerweile scheint's ihm Spaß zu machen) Nein, eine Frage hätte ich noch: Wissen und Glauben : PWS.

ich: Zwei Ansätze zur Modellierung subjektiven Wissens: sentimentelle Semantik und

PWS; behaupte, daß Glauben in einer PWS keinen Sinn macht

Nejdl: Wieso das denn nicht ?

ich: schreibe Knowledge Axiom hin, erkläre Unterschied zwischen Glauben und Wis-

sen

Nejdl: Ja und wieso macht das jetzt keinen Sinn ?

ich: Das hängt von der Erreichbarkeitsrelation ab, ob das Sinn macht.

Nejdl: Bei welcher gilt denn nun das KA ?

ich: Bei einer symmetrischen.

Nejdl: Danke, das reicht.

Nach dem katastrophalen Anfang der Prüfung (SQL und EFRS), bei dem ich aus-

gesehen haben muß wie der letzte Trottel, kamen die Profs beide gegen Ende aus

dem Nicken nicht mehr raus<sup>1</sup> (der Beisitzer meinte nachher, "da hast Du sie aber

zum Schluß echt beeindruckt"). So ist dann wohl auch die doch noch gute Note zu

erklären.

Noch etwas zur Prüfungsatmosphäre: Eigentlich recht nett, aber Jarke hat mich

anfangs mit seinen detailliertesten Schilderungen des Supermarkt-Szenarios ziemlich

jeck gemacht - er mußte alles mehrfach wiederholen, weil er soviel drumherumgere-

det hat, daß ich die tatsächlichen Informationen gar nicht mehr mitbekommen habe

und dadurch noch nervöser wurde. Allerdings sind am selben Tag bis auf einen alle

mit SQL badengegangen, weil keiner so richtig damit gerechnet hatte, daß das so

detailliert drankam. Ich hab mich hinterher nur ein bißchen geärgert, daß ich nicht

noch unaufgefordert etwas über SQL, Tupelkalkül und die Operationen der relatio-

nalen Algebra erzählt habe, aber ich war leider viel zu beschäftigt mit SELECT und

GROUP BY ...

<sup>1</sup>Erstnachherweise war genaue Wissensrepräsentation eigentlich mein Angericht gewesen, während ich mich bei DB und XPS deutlich sicherer gefühlt hatte... was mal wieder bestätigt, daß doch alles anders kommt, als man denkt.

# Prüfungsprotokoll Praktische Informatik

Prüfer : Prof. Dr. M. Jarke  
Fächer : Einf. in DB  
Objekt orientierte DB (Kemper/Morkocette)  
Betriebssysteme (Silberschatz)  
Datum : 20.08.96

- Einf. in DB :
- Im Internet gibt es ein Buchladen, wo man Bücher als Kunde bestellen kann. Ein Kunde kann auch gleichzeitig als Autor sein. Situation in ER-Modell darstellen.
  - ER-Modell in relationalem Modell überführen.
  - Sind die Relation in Normalformen ?
  - Anfrage in SQL formulieren, um alle Kunde herauszufinden, die gleichzeitig Autor sind.
  - ER-Modell in Netzwerkmodell überführen.
  - Wie sehen die Datensätze in Netzwerkmodell aus ?

## OODB:

- Wie sehen die Modellierung von obige ER-Modell in OO-Modell aus ?
- Welche Vorteile hat man in OODB im Vergleich zu relationalen DB ?
- Was muß man bei Vererbung von Operationen beachten ?

## Betriebssysteme:

- Beim Drücken von einem Dokument werden die Daten in Seiten zum Druckerpuffer geschickt, welche Situation tritt da auf ?
- Wie sehen die Lösung von Consumer-Producer Problem mit Semaphor aus ?
- Welche Probleme können da auftreten ?
- Was sind die Deadlock-Bedingungen ?
- Bei welchen konkreten Betriebsmitteln kann kein Deadlock auftreten ? warum ?
- Bei welchen konkreten Betriebsmitteln kann Deadlock auftreten ?
- Was ist Paging-System ?
- Welche Schedulingalgorithmen werden dann verwendet ?



Prüfung:  
 Prüfer: Prof. Dr. M. Jahre  
 Beirater: ?

- Stoff:
- Einführung in Datenbanken (Jahre, WS 92/93)
- Implementierung von Datenbanken (Kemper, WS 91/92)
- Betriebssysteme (Peterson/Silberschatz "Operating System Concepts")
- Datum: 10.6.94
- Dauer: 45 min.



Einführung in Datenbanken:

- Datenbankentwurf in Rahmen einer Informationssysteme → Vergleichsweise (mündliche Konzeptphase) → verschiedene Unterlagen → z.B. Formulare
- Konzept: Eingabedatentyp: Formulare

Fingergeldbeleg		Name		Datum	
Fingergeldbeleg		Name		Datum	
Fing#	von	nach	Preis	Ausgabe	Einnahmen
:	:	:	:	:	:
:	:	:	:	:	:
Gesamteinnahmen: ...					

Voraussetzungen: - 1 Formular / Tag  
 - einleitendes Preis  
 - Anzahl der Folien hängt von  
 - Preis ab

- ER - Modelle aufstellen
- In relationalen Modelle überlegen
- Eine Anfrage in SQL angeben, so daß Folgen  
Anfrage erzeugt wird:

1.12.94.

Frage # Einmal	
	⋮
	⋮
	⋮

- Zusammenhang der Operatoren der relationalen Algebra und der SQL-Anfrage



Implementierung von Datenbanken:

- Wie sieht das 5-Schichten-Modell aus?
- Für die obige SQL-Anfrage: was passiert in jeder Schicht?
- Compiler: Zerlegt die Anfrage in  
Bäume  $\Rightarrow$  Optimierung der Anfrage  $\rightarrow$  genaue Kosten  
Herleitung
- Genaue Beschreibung der Optimierungstechnik
- Wie wird 'select' (g) implementiert?
- Segmentmethode (Ruffa): Unterchied  
zu Virtual Memory

• Was passiert, wenn mehrere Transaktionen gleichzeitig auf Daten zugreifen?

• Wie werden Sperranforderungen geordnet? Gerechtigkeit?

• Baumartige Darstellung der Sperranforderungen (DB, Daten, ..., Satz, Feld).

• Trade-off große/feine Sperranforderungen.

• Wie kann man Sperranforderungen auf dem Baum setzen (Intention Locks)

• Wie funktioniert dies? (Top-Down)

• Komplexität und Anknüpfungspunkte

• zwischen Typen von Sperranforderungen

Betriebssysteme:

• Paging: Definition. Wie erfolgt die Seitenverwaltung (benötigte Hardware und

Vergleichen).

• Wie kann der Gebrauch der Page Table effizient gemacht werden? (Assoziations

Speicher, Cache → ein Teil der Page Table)

• Wie wird Virtual Memory realisiert?

• Funktionsweise von Demand Paging.

• Was passiert wenn die adrethierarchische Seite nicht in Hauptspeicher vorfindet?

• Wie wird eine Seite aus dem Hauptspeicher geholt? (Assoziations

• CPU-Unterstützung, Queuing an I/O-Resourcen, Austausch, ..., Queuing an CPU)



## 5 Literatur

### 5.1 Einführung in Datenbanken

1. Elmasri/Navathe "Fundamentals of Database Systems"  
Deckt die komplette Einführung ab. Ist sehr verständlich geschrieben, mit durchweg guten Beispielen etc. Meine absolute Empfehlung – man braucht eigentlich nichts anderes. Hier lohnt sich ein Kauf in jedem Fall – auch für "später".

2. Kemper/Eickler "Datenbanksysteme", Oldenbourg Verlag, 1996, Bn 5489  
Kompakter als Elmasri/Navathe, vielleicht sogar noch etwas klarer und verständlicher in der Darstellung (da weniger auf Details eingegangen wird). Für den Einstieg (und Lernen unter Zeitdruck) sehr zu empfehlen, nur sollte man sich die Details zu einigen Themen (bes. Normalisierung) dann zusätzlich in Vossen oder Elmasri/Navathe anschauen. Auf Netzwerk/Hierarchische Datenbanken wird nicht eingegangen, was das Buch zwar sympathisch macht – aber Jarke fragt gerne danach, auch wenn der Gral in der Prüfung an mir vorbeigegangen ist :)

3. Vossen "Datenmodelle, Datenbanksprachen ...."  
Laut Jarke steht hier 70% bis 80% des Inhalts der Einf. und Impl. Vorlesungen drin. Was wohl auch stimmt – nach meinem Geschmack ist Vossen allerdings unnötig technisch und formal, daher nicht so verständlich. Nach meiner Meinung nur zum Nachschlagen geeignet.

### 5.2 Implementierung von Datenbanken

Auch hier werden große Teile in den zur Einführung genannten Büchern abgedeckt. Hier noch einige Zusätze:

1. Architektur/Fünf-Schichtenmodell: S. 87–95 und 135–141 und 167–171 aus Lockemann/Schmidt "Datenbank-Handbuch" fotokopieren. Jarke fragt hier sehr gerne nach (siehe andere Protokolle). Zur Prüfungsvorbereitung lohnt es sich auf keinen Fall das Ding zu kaufen, da veraltet !

### 2. Anfragetransformation/Optimierung:

Elmasri/Navathe sowie Kemper/Eickler erzählen wichtige Sachen zur Anfrageübersetzung und logischen Optimierung. Sollte man lesen. Vossen ist hier eher schwach. Zur Tableau-Methode (Semantische Optimierung) gibt es den Original-Artikel Aho/Sagiv/Ullman "Efficient Optimization of a Class of Relational Expressions", ACM Transactions on Database Systems, Vol.4, No.4, Dec 1979. Hier wird die Äquivalenzabbildung etwas besser beschrieben, aber eigentlich reicht die Beschreibung die in der Vorlesung gegeben wird.

Anfrageoptimierung: Nach Parameter-Systemen wird in letzter Zeit oft gefragt. Hierzu sollte man den Original-Artikel Philippe Richard "Evaluation of the size of a Query expressed in Relational Algebra", SIGMOD Conference 1981 lesen, der allerdings ziemlich schlecht ist.

Die Beschreibung des Anfragesystem/R anhand der Vorlesung zu verstehen, halte ich für unmöglich. Die Original-Quelle ist: Selinger et al. "Access Path





*Selection in a Relational Database System*, ACM SIGMOD 1979. Ob es sich wirklich lohnt das durchzulesen weiß ich allerdings nicht – bis jetzt scheint Jarke noch nicht dazu gefragt zu haben.

### 3. Transaktionsverwaltung:

Hier unbedingt die entsprechenden Kapitel im Vossen lesen, da die Folienkopien eine Zusammenfassung hiervon darstellen. Es gibt noch ein spezielles Buch *Vossen/Groß-Hardt "Grundlagen der Transaktionsverwaltung"*, welches einen kleinen Touch ausführlicher ist. Der große Vossen reicht aber auf jeden Fall.

Die Beschreibung des Schatten Speicherkonzeptes ist im Vossen sehr nebulös. Ich empfehle hierzu den entsprechenden Teil im *Korth/Silberschatz "Database System Concepts"*, der allerdings nur den Einbenutzer-Fall erläutert.

Nachdem ich mir die recht formalen Vossen-Definitionen angeschaut habe (die auch gefragt werden), hat mir der entsprechenden Teil in Kemper/Eickler geholfen (insb. zu Recovery) den Gesamtzusammenhang etwas besser zu verstehen und zu sehen welche Probleme in welcher Serialisierbarkeitsklasse/Rücksetzbarkeitsklasse gelöst werden.

## 5.3 Nicht-Standard Datenbanken

1. *Ceri, Gottlob, Tanca "Logic Programming and Databases"*, Springer Verlag 1990, Bm7374. Deckt den kompletten DDB-Teil der Vorlesung ab und ist sehr verständlich und gut lesbar gehalten. Sehr zu empfehlen.

2. *Cremers, Griefahn, Hinze "Deduktive Datenbanken"*, Vieweg Verlag 1994, Bm1699. Hält sich in großen Teilen an CGT90. Ebenfalls gut lesbar. Der Teil über strategierte Negation ist nicht so gut gelungen. Dafür ist die Magic-Set Auswertung besser dargestellt.

Das Standardwerk ist ansonsten *Ullman "Principles of Database and Knowledge-Base Systems"*.

Auf den Teil "Temporale Datenbanken" habe ich mich so gut wie gar nicht vorbereitet. Das war natürlich hoch gepokert, da ich Jarke die Unterlagen zur Vorlesung einreichen musste – somit auch die ausgeteilten Paper von Prof. Sripada über TempDB. Als Jarke mir zu Beginn der Prüfung eröffnete, er hätte gar nich reingeschaut, war ich natürlich froh. Das muß bei der nächsten Prüfung über NSDB natürlich nicht der Fall sein.

Viel Erfolg!



Passagiere: Name, Tel.-Nr.

Flug:

Flug-Nr., Flüge, Startort, Zielort, Fluggesellschaft

Buchen:

Name, Flug-Nr., Datum

(Wieder habe ich alles erzählt was mit dazu einfie)

Jarke: Dann machen wir darauf doch mal eine SQL Anfrage: Ich hätte gern die Namen der Passagiere, die heute nach Frankfurt geflogen sind.

Ich: Erzähle wieder während ich schreibe über: `SELECT=Projektion, FROM=Join, WHERE` mit Join-Bedingung und Select)

SELECT Name

FROM Flug, Buchen

WHERE Flug.FlugNr=Buchen.Flug-Nr (*Join-Bedingung*)

Buchen.Datum=11.03.94

Flug.Zielort="Frankfurt"

Jarke: O.K. die nächste etwas kompliziertere Anfrage: Ich möchte die Anzahl der Passagiere einer jeden Fluggesellschaft, die heute nach Frankfurt geflogen sind.

Ich: `SELECT Fluggesellschaft, COUNT(Name)`

FROM Flug, Buchen

WHERE Flug.FlugNr=Buchen.Flug-Nr

Buchen.Datum=11.03.94

GROUP BY (Fluggesellschaft)

Jarke: Der Join, wie testet man auf Verlustlosigkeit

Ich: Die Tableau-Methode verbal erklärt.

Jarke: Und das heißt Tableau-Methode und wie geht das beim Abschluss der FA's?

Ich: Über die Armstrong Axiome (aufgeschrieben und erklärt s. Skript)

Jarke: Ja gut aber wie geht das denn nun?

Ich: Also theoretisch bildet man den Abschluss über die FD's ( $F^+$ ), aber praktisch bildet man den Abschluss über die Attribute. Der Algorithmus ist induktiv aufgebaut.

Ich: Working-Set-Model erklärt.

Jarke zum Assi: Wann haben wir angefangen?

Assi: Vor etwa 55 min.

Jarke: O.K. dann vielen Dank.

Allgemeines: Jarke ist sehr freundlich und unterbricht selten, er macht aber auch deutlich, wenn er mit irgendwas nicht einverstanden ist. Wenn er eine Frage stellt habe ich ihm zunächst die Schlagwörter an den Kopf geworfen, woraufhin er grinste, beim erklaren dieser Worte verfinsterte sich manchmal sein Gesicht, das darf nicht irritieren. Zum Zustandekommen meiner Note meinte Jarke, daß ich die herkömmlichen Datenmodelle wohl nicht so gut konnte, aber alles andere sehr gut vorbereitet war.

Viel Glück bei Euren Prüfungen auch wenn Ihr es jetzt nicht glaubt aber es geht vorbei Toi, Toi, Toi.



Prüfung: Praktische Informatik  
 Prüfer: Prof. Dr. M. Jahre  
 Stoff: ?  
 Datum: 10. 6. 94.  
 Dauer: 45 min.

• Einführung in Datenbanken (Yake, WS 92/93)  
 • Implementierung von Datenbanken (Kemper, WS 91/92)  
 • Betriebssysteme (Peterson/Silberschatz "Operating System Concepts")



Einführung in Datenbanken:

- Datenbankentwurf in Rahmen eines Informationssystem → Vergleichsweise (mündliche Konzeptphase) → verschiedene Methoden → z.B. Formale
- Konzept: Ringgesellschaft : Formale :

Ringgesellschaft		Name		Datum	
: : : : : :		: : : : : :		: : : : : :	
Flug#	von	nach	Preis	Ausschlag	Einnahmen
Gesamteinnahmen: : : : : :					

Voraussetzungen: - 1 Formular / Tag  
 - einheitlicher Preis  
 - Anzahl der Flüge hängt von Flug ab

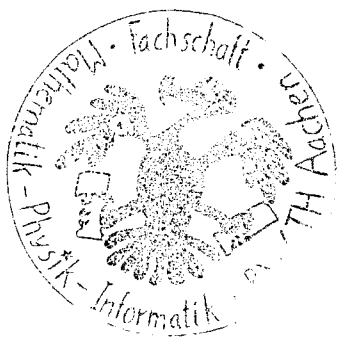
- Implementierung von Datenbanken:
- Wie nicht den 5-Schichten-Modell aus?
- Für die obige SQL-Anfrage: was passiert in jeder Schicht?
- Generator: erstellt: Darstellung der Anfrage als Baum  $\Rightarrow$  Optimierung der Anfrage  $\rightarrow$  genaue Kosten  $\rightarrow$  Hardware
- Genaue Beschreibung der Optimierungshierarchie
- Wie wird 'SETC' (G) implementiert?
- Segmentierte (RPGA): Unterchied zu Virtual Memory

Zusammenhang der Operationen der relationalen Algebra und der SQL-Anfrage

Flug#	Einwahl
:	:
:	:

1.12.94.

- ER-Model aufbauen
- In relationalen Modelle äquivalente
- Eine Anfrage in SQL angeben, so daß folgende Ausgabe erzeugt wird:



Was passiert, wenn mehrere Transaktionen gleichzeitig auf Daten zugreifen?

Wie werden Sperrn gesetzt? Granularität?

Baumartige Darstellung der Sperrlichkeiten

(DB, Befehle, ..., Satz, Feld)

Trade-off große/feine Sperrn

Wie kann man Sperrn auf der Baum setzen

(Intaktion Locks)

Wie funktioniert dies? (Top-Down)

Komplexität und Abhängigkeiten

zuerst Typen von Sperrn

Definition: Wie erfolgt die Sperrung

(benötigte Hardware und Software)

Vergleichsweise

Wie kann der Gebrauch der Page Table effizient gemacht werden? (Assoziativen Speicher, Cache → ein Teil der Page Table)

Wie wird Virtual Memory realisiert?

Funktionsweise von Demand Paging

Was passiert wenn die adrezierte Seite nicht in Hauptspeicher befindet?

Wie wird eine Seite aus dem Hauptspeicher geladen? (Assoziativen Speicher)

günstigeren (..., Queueing an I/O-Processor, ..., Queueing an CPU)



Fächer: Datenbanken und Informationssysteme I  
Objektorientierte Datenbanken  
Wissensbasierte Informationssysteme  
Implementierung von Datenbanken

Prof. Jarke  
Prof. Unland  
Prof. Hahn  
Prof. Jarke

Es wurden folgende Fragen gestellt:

1. Objektorientierte Datenbanken

- Worin bestehen die Vorteile gegenüber den konventionellen Datenbanksystemen nach dem Relationenmodell?
- Wie könnte man einen Tupelwert für ein Attribut im relationalen Modell darstellen? (Beispiel Eltern - Kinder)
- [Von jedem Kind einen Zeiger auf die Eltern.]

2. Datenbanken und Informationssysteme I

- Wodurch kann sowohl der direkte als auch der sequentielle Zugriff schnell abgearbeitet werden? [B\*-Baum]
- Wie wird der B\*-Baum aufgebaut?
- Wie kann man ihn optimieren?
- Wie errechnet sich seine Tiefe?

3. Implementierung von Datenbanken

- Beschreiben Sie die Arbeitsweise des Join-Operators!
- Wie kann man ihn implementieren?

4. Wissensbasierte Informationssysteme

- Beschreiben Sie die Bestandteile eines Expertensystems.
- Worin bestehen die einzelnen Funktionen?
- (Als Grundlage wurde MYCIN verwendet.)

Die Atmosphäre in der Prüfung ist sehr angenehm. Es besteht keine Hektik und man hat so genügend Zeit zum Überlegen.



Handwritten initials 'MH' in a circle.



WHERE Patient.Nr = IBAREL.PNR and IBAREL.KNR = Krankheit.Nr and  
Krankheit.Name = "Schnupfen"  
Zweite Loesung Man geht von Krankheit aus und zaehlt nur die  
Beziehungen, der Patient ist egal, und vor allem hat ein Patient nicht  
zweimal Schnupfen (laut Jarke, ich dachte da so an das Datensattribution,  
doch das war ja sowieso konstruiert)

Pf: Aehn (an den Besitzer) Haben wir BS schon gepurvt (natuerlich  
noch nicht, aber man merkte Jarke an, dass er keine Lust mehr  
hatte) Haben wir denn da ausgemacht, da"s wir ueber  
Sekundaerspeicherverfahren pruefen koennen.  
Ich: (Ich denk mir, sag bloss nichts falsches, sonst fragt er Dich  
Sachen, die du noch nicht mal gelesen hast, obwohl BS mein  
Lieblingsthema ist) Wir nur ueber Kap 1-9 + Fallbeispiel UNIX, also  
keine Filesystems und Protections.  
Pf: Koennen Sie mir den etwas ueber Verfahren der Sektoranforderung  
sagen ?  
Ich: Ja, FCFS (FIFO), SCAN, C-SCAN, LOOK, C-LOOK. Toner an der ganzen  
Sache ist halt Kopfbewegung.  
Pf. Gut, aehmmm (dann zum Besitzer) Haben Sie noch etwas, was Sie  
fragen moechten ?

Beisitzer: Was koennen Sie fuer Verfahren den Speicher zu vergr"o"sern?  
Ich: (Komische Frage, aber ich weiss was gemeint ist) Sie sprechen  
virtuelle Speicherverwaltung an. Alsoooo : Paging, Segmentation und  
Stichworte dazu : interne / externe Fragmentierung. Ohh da faellt  
mir noch ein aelterer Ansatz ein: Overlays.  
Beisitzer: Erklaeren Sie mal Segmentation ?  
Ich: (Auf so warte ich ja nur) Grob skizziert, wie und wer teilt  
auf. "Demand" dabei erlaert.

Beisitzer: Koennen Sie konkrete Systeme in denen so vorgegangen wird.  
Ich: Ja OS/2 fuer Segm. (ich hasse INTEL / IBM, obwohl besser als garnichts  
(MSDOS) und UNIX fuer Paging, sowie alle moderneren Systeme  
Beisitzer: Was mu"s die Hardware bieten, damit man mehrerer Prozesse  
laufen lassen kann.

Ich: (Haeehh, wie kommt der jetzt darauf, war wohl auch so aus dem  
Bauch heraus) Aeh, Bei Time-sharing Interrupteigenschaften und  
allgemein die Dual-Mode operation ?? (Beisitzer schien zufrieden)  
Beisitzer: Wie lautet denn so ein Page-fault ab, was macht die Hardware  
dabei ?  
Ich: Zugriff --> Adressfehler (Exceptioneigenschaften des Prozessors)  
--> BS entscheidet ob Pagefault oder Adressfehler usw.

Pf: Wir machen jetzt was fruherer Schluss, weil ich gleich noch einen  
Pruefiling mehr als vorgesehen habe.

ENDE

Allgemein: Pf ist ruhig, draengelt nicht und laesst einen ausreden.  
Auch wenn er keine Zeit hat ist er mit Fragen und Erklarungen nicht  
der schnellste. Dies kann vorteilhaft oder von Nachteil sein. Ich  
haette lieber mit DB oder BS angefangen, aber er hat mich nicht  
gefragt, was er sonst aber anscheinend macht. Das ergab ein bisschen  
Zeitnot da ich bei CB mich etwas lange aufgehalten habe (laut Jarke  
war die Grammatik der Grund fuer 1.7 (was solls)). Er liest  
irgendein Kapitel was dann zu seinem Lieblingsthema fuer den Tag wird.  
Es empfiehl sich, wenn moeglich, vorher einlige Pruefilinge abzufragen.  
Alles in allem nett !

Gedächtnis-Protokoll einer Diplomprüfung Praktische Informatik

Prüfer: Prof. Jarke

Termin: 15. Februar 1993

Fächer und Referenzen: Betriebssysteme (Silberschatz, Peterson, Galvin: Operating System Concepts), Datenbanken (Frei nach Jarke), Wissensrepräsentation (Geneserth, Nilsson: Logische Grundlagen der KI)

Dauer: ca. 55 min.

Note: 1.0

**Datenbanken**

Jarke: Definition der Prädikatenlogik erster Stufe?

Ich: Definition in pseudo-EBNF hingeschrieben.

Jarke: Wo spielt die PL1 bei den Datenbanken eine Rolle?

Ich: Relationales Kalkül: Tupel- und Domänenvariante erwähnt.

Jarke: Wie kann man das relationale Kalkül definieren?

Ich: Induktive Definition von Anfragen im Tupel-Kalkül aufgeschrieben.

Jarke: Wie unterscheiden sich Tupel- und Domänenkalkül bezüglich der PL1?

Ich: In den Wertebereichen der Variablen.

Jarke: Wie kann man Anfragen des relationalen Kalküls durchführen?

Ich: Übertragung in relationale Algebra.

Jarke: Wie ist die definiert?

Ich: Definition der Grundoperationen sowie einiger darauf aufbauender Operationen aufgeschrieben.

Jarke: Und wie sieht die Übertragung des Kalküls in diese Algebra aus?

Ich: (Geriet etwas ins Schwitzen.)

Jarke: Nur exemplarisch: also stellen Sie sich vor... (murmelte etwas von „Existenzquantoren sollten schon drin sein“) und malte zwei Relationenschemata mit einem gemeinsamen Attribut auf. Wie sieht die Anfrage (irgendeine einfache Anfrage mit einem Join) aus? Erstmals in SQL.

Ich: Hingeschrieben.

Jarke: Und wie formuliert man das im Tupelkalkül?

Ich: Auch hingeschrieben.

Jarke: Und wie übertägt man diese Anfrage in die Algebra?

Ich: Grob skizziert: Konjunktion von Existenzquantoren in kartesische Produkte, Bedingungen an existenzquantifizierte Variablen in Selektion, Bedingungen an die Anfragevariable in Projektionen etc. Resultat nochmal hingeschrieben.

Jarke: Wo spielt die PL1 bei den Datenbanken noch eine Rolle?

Ich: (Grübel, grübel) Bei den Constraints?

Jarke: Ja, Constraints kann man als PL1-Formeln auffassen: aber wie formuliert man Constraints bei beim Datenbankenwurt?

Ich: (Mit einigen Hilfestellungen und Tips) Mit funktionalen Abhängigkeiten.

Jarke: Wie ist die funktionale Abhängigkeit definiert?

Ich: Definition hingeschrieben.

Jarke: Wo treten die Abhängigkeiten beim Entwurf auf? Also, welche Eigenschaften sind für einen Entwurf wünschenswert?

Ich: Verlustlosen Join und Erhaltung der funktionalen Abhängigkeiten definiert; Gewährleistung von Redundanzfreiheit durch Normalformen erwähnt.

Jarke: Wie ist der Abschluss (den ich bei der Definition der Abhängigkeitserhaltung erwähnt hatte) definiert?

Ich: Als Abschluss unter den Armstrong-Axiomen.

Jarke: Wie lauten die?

Ich: Erstes Axiom hingeschrieben.

Jarke: (Unterbricht:) Na, dann glaube ich Ihnen mal, daß Sie die anderen auch können. Wie beweist man die Axiome?

Ich: Direkt mit der Definition der funktionalen Abhängigkeit.

Jarke: Welche Verfahren zum Datenbankenwurt gibt es?

Ich: (Nachdem ein Ansatz mit Normalformen abgewurt wurde:) Komposition und Dekomposition von Relationenschemata.

Jarke: Wie verhalten sich die Verfahren bezüglich der erwünschten Eigenschaften?

Ich: Die Komposition erhält die funktionalen Abhängigkeiten; die Dekomposition den verlustlosen Join.