Zusammenfassung

VL Big Data Engineering
(aka Informationssysteme)

Prof. Dr. Jens Dittrich

bigdata.uni-saarland.de

16. Juli 2020

Behandelte Anwendungen in der Vorlesung

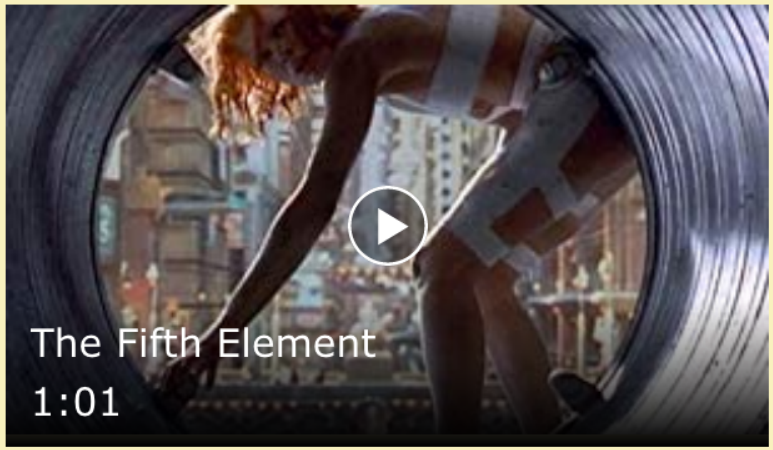
1. **IMDb**
2. NSA
3. Grundlagen: Anfrageoptimierung
4. Handel, Banken, Ticketsystem

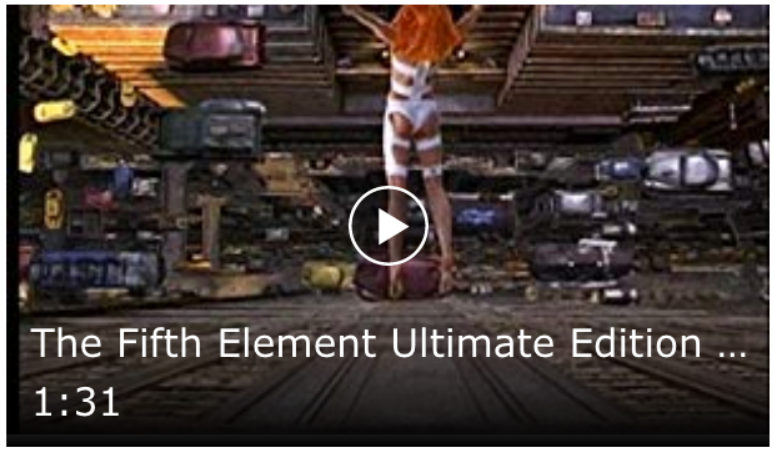
IMDb

the fith element

All

The Fifth Element (1997)
Bruce Willis, Milla Jovovich





The Fifth Element (1998)
Action, Adventure, Sci-Fi

Elementary: Season 5 - The Fifth Elementary (2017)
Nelsan Ellis, Jon Michael Hill

The Fifth Element (2002)

Beatboxing: The Fifth Element of Hip Hop (2011)

Shazam! Soars
Top
Weekend Box Office

Browse trailers »

Zachary Le

Anwendung: IMDb

Frage 1

Wie werden in IMDb die Daten zu Filmen, Schauspielern, Regisseuren usw. modelliert und abgelegt?

Modellierung: Entity-Relationship-Modell, relationales Modell

Datenablage: Dies ist ein Freiheitsgrad, über den wir im Moment keine Entscheidung treffen müssen: jede Form der Datenablage, die Konform ist mit dem Relationalen Modell, kann gewählt werden!

Diese Eigenschaft nennt man **Physische Datenunabhängigkeit**.

IMDb

Frage 2

Wie werden in IMDb Verknüpfungen dieser Daten modelliert und abgelegt?

durch Fremdschlüsselbeziehungen im relationalen Modell

IMDb

Frage 3

Wie stellen wir Anfragen an diese Daten?

Mit der relationalen Algebra!

Achtung

Die Relationale Algebra beschreibt nur abstrakt **WAS** berechnet werden soll, aber nicht **WIE** diese Berechnung algorithmisch umgesetzt wird!

Leider sind Ausdrücke in relationaler Algebra manchmal etwas unübersichtlich. Deswegen ist es eine andere Option, die relationale Algebra unter einer anderen Sprache zu verstecken, d.h. eine andere Anfragesprache zu konzipieren und diese dann jeweils automatisch in die relationale Algebra zu übersetzen.

Fundamental Theorem of Software Engineering (FTSE):

„We can solve any problem by introducing an extra level of indirection . . . except for the problem of too many levels of indirection."
[David Wheeler]

Wichtigste Lernziele

Entity-Relationship-Modellierung

ER, Entitätstypen, Beziehungstypen (auch mehrstellig), Domäne, Attribut, Schlüssel, Funktionalitäten (Chen-Notation), Funktionale Bestimmtheit, N:M, 1:N, 1:1, N:M:P, Generalisierung

Relationales Modell

Relation, Tupel, Relationenschema, Umsetzung von ER nach RM, Schlüssel, Fremdschlüssel

Wichtigste Lernziele

Relationale Algebra

Operatoren (unär und binär), Selektion (nicht zu verwechseln mit SELECT in SQL), Projektion, Vereinigung, Differenz, Kreuzprodukt, Schnitt, Theta Join, Gruppierung vs Aggregation

Behandelte Anwendungen in der Vorlesung

1. IMDb
2. **NSA**
3. Grundlagen: Anfrageoptimierung
4. Handel, Banken, Ticketsystem

Die Spionageaffären

- seit Bestehen von Geheimdiensten gab es immer wieder Whistleblower (Dt.: Enthüller, Aufdecker, Informant, -innen)
- der bekannteste ist Edward Snowden, der bis Mai 2013 als Sysadmin bei der NSA arbeitete
- ab Juni 2013 hat er angefangen, nach und nach geheime Dokumente der NSA zu veröffentlichen, die die Massenüberwachung durch die Geheimdienste dokumentieren
- https://de.wikipedia.org/wiki/Edward_Snowden
- andere wichtige Whistleblower waren Martin und Mitchell, William Binney, Russ Tice, Thomas Tamm und Thomas Drake

Laura Poitras / Praxis Films CC-BY-SA 3.0

NSA: Nationales Sicherheitsamt, Andreas Eschbach

Buchidee

Stellen Sie sich vor, die Computertechnologie hätte sich 70 Jahre eher entwickelt. In der Weimarer Republik gab es bereits Komputer, das Weltnetz und später mobile Volkstelefone. Und umfangreiche Datensammlungen. Dieser Datenschatz fällt bei der Machtergreifung den Nazis in die Hände. Was hätte dies für Auswirkungen gehabt?

- brilliante Buchidee
- Datenanalyse wird in dem Buch zu 95% technisch korrekt beschrieben bis hin zu Beispielen in „Strukturierter Abfrage-Sprache“
- Link zum Buch

NSA

Frage 1

Wie stellen wir so komplexe Anfragen?

SQL!

Häufigste Fehler im Umgang mit SQL 1/3

„SQL ist eine Sprache für das Schreiben und Lesen einzelner Tupel.“

⇒ „Ich nehme SQL hauptsächlich für das Lesen und Schreiben einzelner Tupel: CRUD (Create, Read, Update, Delete). D.h. eine Art tupelartiges Dateisystem“.

Das ist ungefähr so, als würde ich eine komplette Fabrikproduktionsstraße nur als Flaschenöffner benutzen.

- Die wahren Stärken von SQL bleiben so ungenutzt.
- Funktionalität, die eigentlich in SQL vorhanden ist, wird nachimplementiert, mit allen (versteckten) Kosten: Qualitätssicherung, Testen, Bug Fixes, ...

Häufigste Fehler im Umgang mit SQL 2/3

„SQL und insbesondere Joins sind langsam.“

⇒ „Ich nehme lieber NoSQL, Hadoop oder implementiere es selbst“.

- Bitte geben Sie Ihren Bachelor bei unserer Studienkoordinatorin zurück.
- SQL und die Performance von SQL-Statements sind **zwei verschiedene Dimensionen**
- die Performance von SQL hängt von sehr vielen Faktoren ab, aber nicht von prinzipiellen Limitierungen von SQL!
- die wichtigsten Einflussfaktoren: Indexe, Art der (Anfrage-)Optimierung von SQL, Physisches Design
- dazu später und in der Stammvorlesung mehr

Häufigste Fehler im Umgang mit SQL 3/3

„SQL kann nicht mit stärker strukturierten Daten wie JSON, Objekten, Graphen umgehen“

⇒ „Ich nehme lieber einen Key/Value-Store“.

- bereits für SQL 1999 wurde das relationale Modell erweitert
- Grundidee: Domänen können beliebigen Typs (insbesondere strukturiert!) sein und nicht nur „atomare Typen“
- rich datatypes: arrays, nested tables, composite types, ..
- SQL 2016: JSON
- gutes Übersichtsvideo hierzu: Markus Winand, The Mother of all Query Languages: SQL in Modern Times

Seit SQL-92 ist sehr viel passiert...

Aber in vielen Projekten wird nur SQL-92 oder wenig mehr benutzt.
D.h. es wird oft viel Potential und Geld verschwendet.

NSA

Frage 2

... und welche ethischen Probleme entstehen durch diese Anfragen? Wie gehen wir damit um?

- ziviler Ungehorsam
- technische Maßnahmen
- soziale Maßnahmen
- rechtliche Maßnahmen

Wichtigste Lernziele

SQL

Kernidee, häufigste Fehler, Grundstruktur einer SQL-92-Anfrage, konzeptuelle Ausführungsreihenfolge, Joins, Gruppierung, Outer Joins, HAVING, Unteranfragen, (dynamische) Sichten

Big Data-Arithmetik

Äpfel + Birnen = whatever; mit anderen Worten: der Informationsgewinn, der durch das Joinen zweier Datensätze entstehen kann, ist in der Regel unvorhersehbar

Big Data: Gegenmaßnahmen

Datensparsamkeit, Anonymisieren, Verschlüsseln, Verrauschen von Daten (Differential Privacy), Anti-Hardware-Administration, Anti-Software-Engineering, Soziale Ansätze, Rechtliche Ansätze

NSA

Frage 3

... und was ist, wenn die Daten größer werden? Wie kommen wir eigentlich von SQL zu einem effizienten Programm?

automatische Anfrageoptimierung

Behandelte Anwendungen in der Vorlesung

1. IMDb
2. NSA
3. **Grundlagen: Anfrageoptimierung**
4. Handel, Banken, Ticketsystem

Was mache ich, wenn die Anfragen zu langsam sind?

Frage 1

Wie kommen wir eigentlich prinzipiell von SQL zu einem ausführbaren Programm?

Automatische Anfrageoptimierung

Frage 2

Was sind die typischen Performance-Probleme dabei und wie kann ich sie lösen?

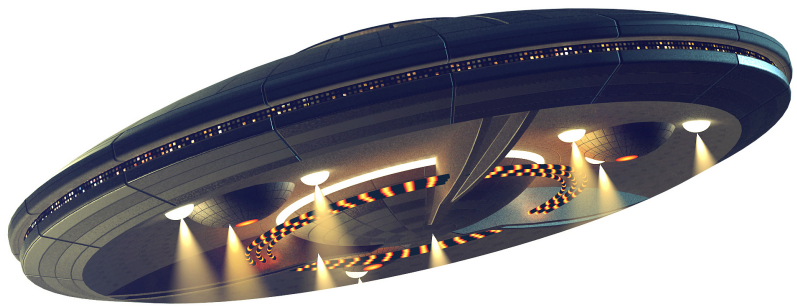
Indexe, Pipelining, Multi-Threading, mehr hierzu in der Stammvorlesung „Database Systems“

Übersicht über SQL-verarbeitende Systeme (SQL-Engines)

Sqlite

MySQL

PostgreSQL



Moderne Datenbanksysteme

Automatische Anfrageoptimierung

1. SQL
   ↓ kanonische Übersetzung
2. annotierte relationale Algebra/logischer Plan
   ↓ heuristische Optimierung
3. transformierter logischer Plan
   ↓ kostenbasierte Optimierung
4. physischer Plan
   ↓ Code-Erzeugung
5. ausführbarer Code

Alle diese Optimierungen laufen intern ab: der Nutzer einer SQL-Engine muss sich **nicht** darum kümmern.

Je nach Anfrageoptimierer können einige dieser Übersetzungsschritte sehr simpel ausfallen, z.B. mit sehr einfachen Heuristiken oder auch ohne kostenbasierte Optimierung.

Wichtigste Lernziele

Automatische Anfrageoptimierung

Grundsätzliche Funktionsweise, kanonische Übersetzung, heuristische Optimierung, kostenbasierte Optimierung, Code-Erzeugung

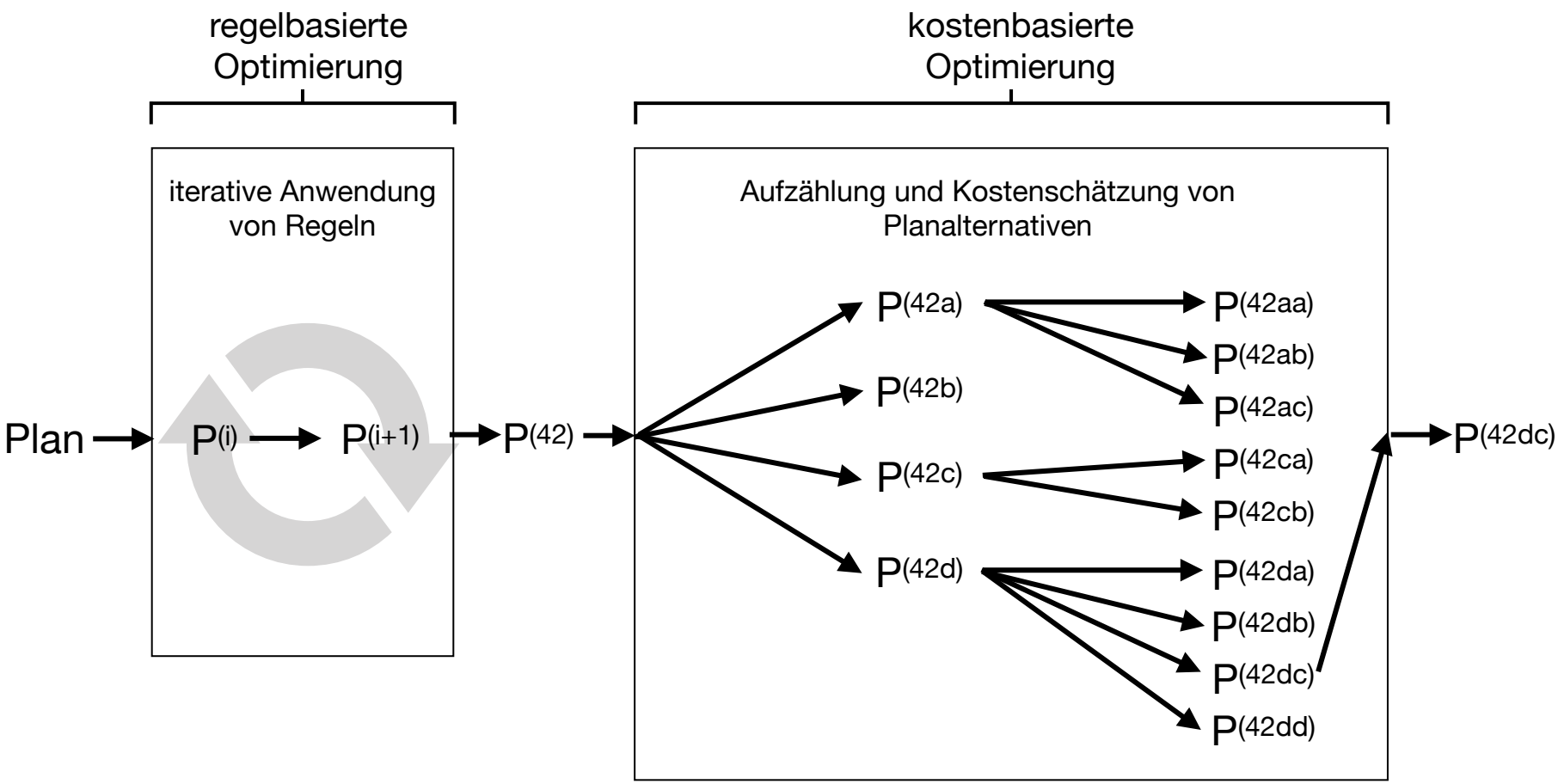
Physische Operatoren

logische vs physische Operatoren, scan-basiert, nested-loops, hash-basiert

Heuristische Optimierung

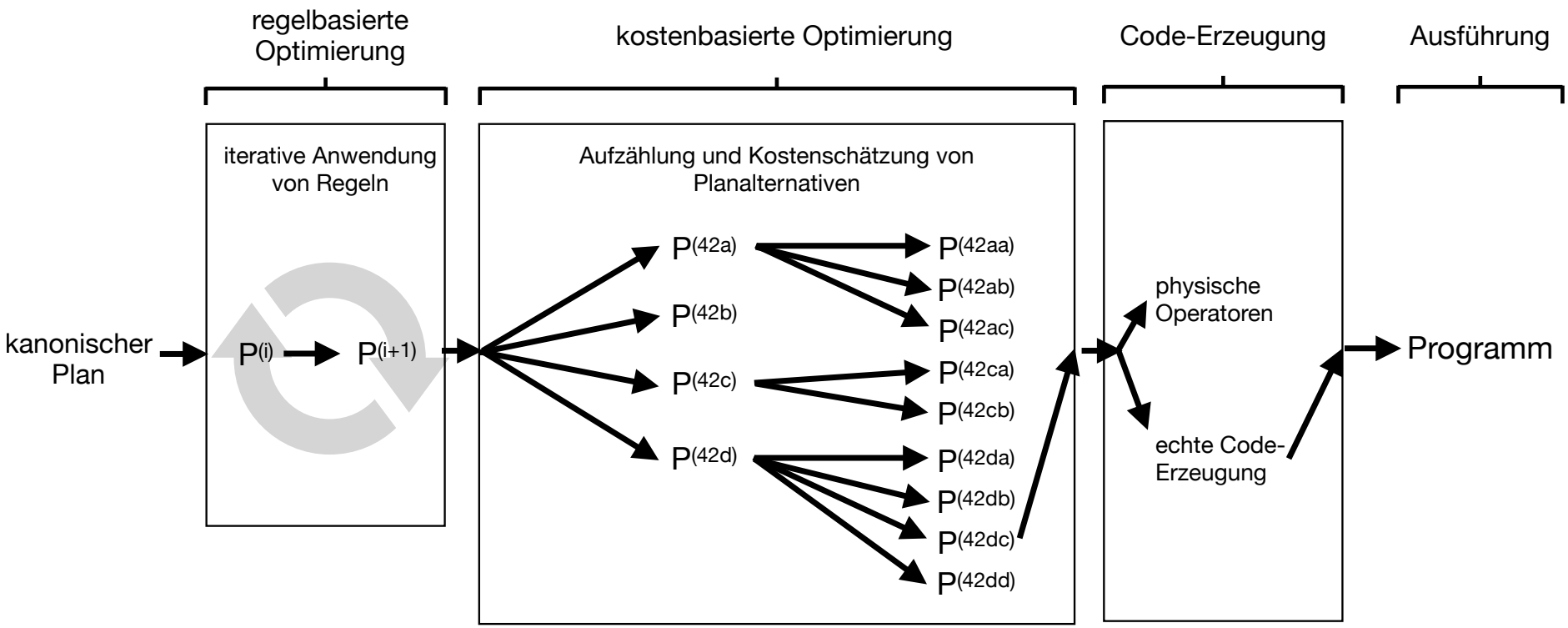
Regeln, Grundalgorithmus, iterative Anwendung von Regeln, Predicate Pushdown, Joinprädikate und Kreuzprodukte zu Joins

Unterschied regelbasierte vs kostenbasierte Optimierung



- in **beiden** Komponenten werden Pläne mit Regeln erstellt
- Unterschied: Anwendung von Regeln **ohne oder mit Kostenschätzung**

Optimierung, Code-Erzeugung und Ausführung: Übersicht



Wichtigste Lernziele

Kostenbasierte Optimierung

Selektivität, Kostenmodelle (Zwischenergebnisse, I/O-Kosten, Gesamtlaufzeit, Energieverbrauch), Modell vs Realität, Plan-Diagramme (Picasso-Diagramme)

Verschiedene Dimensionen kostenbasierter Optimierung

Entscheidungen auf **logischer** Ebene:

1. Welche Joinreihenfolge nehmen?
   Beispiele: Welche Joinreihenfolge hat geringere Laufzeit: $R \bowtie (S \bowtie T)$ oder $(R \bowtie S) \bowtie T$ oder ...?

Entscheidungen auf **physischer** Ebene:

1. Welchen physischen Operator nutzen?
   Beispiele: Hash-basierter Join, sortierbasierter Join oder XY Join? (vgl. Logische vs physische Operatoren-Diskussion)
2. Index benutzen oder nicht? Falls ja, welchen Index benutzen?
   Beispiele: Relation Scannen oder Binäre Suche nutzen? Welche Art von Index nutzen? hash-basiert, baumbasiert oder ... ?
   (vgl. Picasso-Notebook)
3. Welche Ressourcen für welchen Teilplan nutzen?
   Beispiele: Wieviele Threads wo einsetzen? Welcher Teil des Plans bekommt wie viel Rechenzeit/Hauptspeicher?

Wichtigste Lernziele

Joinreihenfolge

Baumstruktur des Plans, left-deep, right-deep, bushy, Reihenfolge der Eingaberelationen

Planvarianten

Join-Selektivität, Join-Graph, Kommutativität von Plänen

Pipelining

HashJoin-Algorithmus, Materialisierung von Zwischenergebnissen, Pipeline-Breaker, Planabschnitte

Physische Optimierungen

Plan-Interpretierung vs Plan-Kompilierung, Horizontale Partitionierung und Multi-Threading, Beispiel in Spark

Behandelte Anwendungen in der Vorlesung

1. IMDb
2. NSA
3. Grundlagen: Anfrageoptimierung
4. **Handel, Banken, Ticketsystem**

ONLINE-BANKING

Zweite IT-Panne innerhalb kurzer Zeit – Commerzbank verärgert Kunden

Viele Commerzbank-Kunden haben am Freitag kein Online-Banking machen und kein Geld abheben können. Erst vor wenigen Wochen gab es einen ähnlichen Vorfall.

Andreas Kröner

24.08.2018 • Update: 24.08.2018 - 17:58 Uhr • Kommentieren • 2 x geteilt

22. Mai 2019, 05:11 Uhr Finanzindustrie

Software-Panne bei der Deutschen Bank

- Die Deutsche Bank hat eine IT-Panne bei einer Software entdeckt, mit der eigentlich Zahlungen von Großkunden überwacht werden sollten.
- Insidern zufolge waren bei diesem Programm jahrelang Parameter falsch programmiert.
- Die Deutsche Bank hat das Problem der Finanzaufsicht Bafin sowie der US-Notenbank Fed gemeldet. Man arbeite daran, "den Fehler schnellstmöglich zu beheben".

Technikpanne

Sparda-Bankautomaten ausgefallen - 3,6 Millionen Kunden betroffen

Wegen einer technischen Panne konnten Sparda-Kunden stundenlang kein Geld mehr den Filialautomaten ziehen. Auch Onlineüberweisungen funktionierten nicht. Sogar die Telefone des Instituts waren ausgefallen.

Logo der Sparda-Bank (Archiv)

Teilen Twittern E-Mail

herstag, 17.05.2018 19:00 Uhr Drucken Nutzungsrechte Feedback Kommentieren

Online-Bank N26

Pannen bei gefeierter Online-Bank

Von Barbara Schäder - 10. April 2019 - 19:18 Uhr

Die Smartphone-Bank N26 hat in nur drei Jahren zweieinhalb Millionen Kunden erobert. Doch nun hagelt es Kritik. Auch die Finanzaufsicht Bafin soll Verbesserungen angemahnt haben.

PROBLEME BEI ZAHLUNGSAUFTRÄGEN

Ärger über IT-Panne bei Commerzbank

VON HANNO MUSSLER - AKTUALISIERT AM 04.06.2019 - 14:53

Eine IT-Panne sorgt für Ärger bei Kunden der Commerzbank. Wegen einer technischen Störung konnten am Montag Daueraufträge, Überweisungen und Lastschriften nicht verarbeitet werden.

Datenverlust

Gravierende IT-Panne bei der UBS – bis zu 1500 Kunden betroffen

Mi 05.12.2018 - 10:01 Uhr | Aktualisiert 05.12.2018 - 10:01
von beat schmid, ch media

Ein IT-Fehler hat bei der UBS zu einer Datenpanne geführt. Dokumente von bis zu 1500 Kunden könnten verloren gegangen sein.

09.01.2019 | Unternehmen

Wieder IT-Probleme bei der Bank Austria

Nicht umsonst nimmt die FMA den Umgang der Banken mit der IT-Sicherheit 2019 besonders unter die Lupe. Die Dringlichkeit verdeutlicht ein schwerer Vorfall bei der Bank Austria.

Handel, Banken, Ticketsystem

Frage 1

Wie erlauben wir das nebenläufige Ändern von Daten, ohne dass fehlerhafte Daten entstehen?

Nebenläufigkeitskontrolle (Concurrency Control)

Frage 2

Wie entwerfen wir das so, dass das Verfahren und das resultierende Gesamtsystem effizient sind?

Beispielsweise durch S2PL mit Prädikatsperren bzw. abgeschwächte Garantien mittels Isolationsstufen.

Moderne DBMS benutzen Multi-Version Concurrency Control (MVCC) oder eine Kombination mit S2PL (siehe Stammvorlesung).

Wichtigste Lernziele

Datenbankmanagementsysteme (DBMS)

Was zeichnet ein DBMS aus?

Datenbankmanagementsysteme (DBMS)

Seit den 70er Jahren werden relationale Datenbanksysteme (RDBMS, meist nur DBMS) entwickelt. Moderne DBMS verfügen typischerweise über folgende Eigenschaften:

1. erweitertes relationales Model: JSON, Arrays, Text, räumliche Daten, etc.
2. sehr umfangreicher SQL-Dialekt (je nach System)
3. **automatischer** Anfrageoptimierer (regel- und kostenbasiert)
4. sehr hohe Performanz (meistens, je nach System)
5. massive Unterstützung für physisches Design (Indexstrukturen, Partitionierung, Caching, Materialisierung)
6. Unterstützung „moderner“ Hardware: DRAM, PRAM, FPGAs, GPUs, ...

Datenbankmanagementsysteme (DBMS)

Und was das Vermeiden und den Umgang mit Fehlersituationen angeht insbesondere:

7. umfangreiche Unterstützung für Transaktionen und ACID
8. Concurrency Control (**automatische** Nebenläufigkeitskontrolle)
9. **automatische** Crash-Recovery, Replikation, Backup
10. **automatische** Konsistenzkontrolle (leider oft nicht genutzt)
11. dynamische Sichten, Zugriffsrechte (logische Datenunabhängigkeit)

Wichtigste Lernziele

Transaktionen

Lese- vs Schreiboperationen, ACID: Atomarität, Konsistenz und Konsistenzbedingungen, Isolation, Dauerhaftigkeit, Automatisierung der Nebenläufigkeitskontrolle

Serialisierbarkeitstheorie

Ausführungsplan (Historie, Schedule), Konfliktoperationen, AP mit totaler vs partieller Ordnung, serieller AP, konfliktäquivalent, konfliktserialisierbar, Vertauschen von Nicht-Konfliktoperationen, Konflikt-Graph, Zyklus, Konfliktserialisierbarkeit im Konflikt-Graphen, Erstellen eines zyklenfreien Konflikt-Graphen, AP vs physischer Plan, Batch vs Operation-at-a-time

Wichtigste Lernziele

Two-Phase Locking (2PL)

Sperren, Lesesperre (shared lock, R oder S), Schreibsperre (exclusive lock, X), Kompatibilität von Lese-/Schreibsperren, 2PL vs Konfliktserialisierbarkeit, kaskadierendes Rücksetzen, striktes 2PL (S2PL), Phantomproblem, Prädikatsperren, Deadlocks, konservatives 2PL (C2PL), Dirty Reads, Lock Ordering

Isolationsstufen

Abschwächung der Isolationsgarantien, `READ UNCOMMITTED`, `READ COMMITTED`, `REPEATABLE READ`, `SERIALIZABLE`, der Default ist meist **nicht** `SERIALIZABLE`!, Auswirkungen auf Anwendungen

Zusammenfassung (Serialisierbarkeitstheorie und Isolationsstufen)

Konfliktserialisierbar $\Rightarrow$ Serialisierbarkeit

Konfliktserialisierbarkeit impliziert Serialisierbarkeit im Sinne der obigen theoretischen Definition: Der Ausführungsplan ist äquivalent zu einem seriellen Ausführungsplan. Mit anderen Worten: die Verschränkung der einzelnen Operationen im Ausführungsplan ist kein Problem.

Das Problem damit: was, wenn nicht alle Transaktionen im Ausführungsplan committen?

Deswegen müssen wir mehr machen:

Konfliktserialisierbar $\not\Rightarrow$ `Serializable`

Die Isolationsstufe `Serializable` ist viel stärker als das theoretische Konzept und verhindert auch Probleme, die durch abbrechende Transaktionen auftreten können.

Zusammenfassung (Serialisierbarkeitstheorie und Isolationsstufen)

Isolation

Die meisten Datenbanksysteme garantieren Atomarität und Isolation für Transaktionen. Durch nebenläufige Ausführung von Transaktionen wird die Leistung dieser Systeme enorm erhöht, ohne dadurch Probleme zu erzeugen.

Vorsicht mit Isolationsstufen

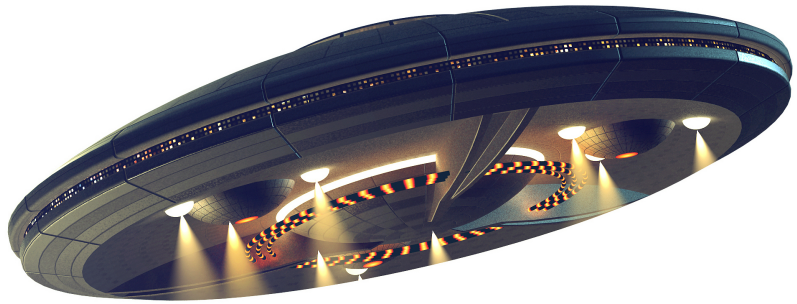
Isolationsstufen sind teilweise schwierig zu interpretieren. Im Zweifel immer die stärkere Isolationsstufe nutzen! Überprüfen Sie immer, welche Isolationsstufe per Default eingestellt ist und was das konkret im genutzten Datenbanksystem bedeutet!

Behandelte Anwendungen in der Vorlesung

1. IMDb
2. NSA
3. Grundlagen: Anfrageoptimierung
4. Handel, Banken, Ticketsystem

Ausblick Stammvorlesung

PostgreSQL



Moderne Datenbanksysteme

Fragestellungen:

- Wie funktioniert eine Flugzeugturbine im Detail?
- Wie fliegen wir mit hundertfacher Lichtgeschwindigkeit?
- Gibt es Wurmlöcher? Und wie fliegen wir da durch?
- Wie können wir die Sicherheit der Passagiere garantieren? Selbst im Falles eines Absturzes?
- Wie bauen wir ein verteiltes Datenbanksystem von der Größe eines Planeten, einer Galaxie, des Universums?

TutorInnen für Big Data Engineering 2021

Kriterien

Falls Sie:

1. in der Abschlussklausur mindesten eine 1,7 erreichen,
2. ein Talent dafür haben, technische Inhalte zu erklären und
3. Lust haben, 2021 bei uns im Team mitzumachen,

Dann: bewerben Sie sich direkt bei mir.

Die Vorlesung wird sich 2021 inhaltlich nicht wesentlich von 2020 unterscheiden (Modulo kleinere Verbesserungen hier und da). Allerdings habe wir voraussichtlich wieder ein Semester in voller Länge und damit zwei Wochen mehr Stoff.

Aufgaben

1. Halten eines Tutoriums,
2. Teilnahme an wöchentlicher Teambesprechung,
3. (Mit-)Konzeption von Übungsaufgaben,
4. Korrektur von Übungsaufgaben und Klausuren.

BSc/MSc „Data Science and Artificial Intelligence“

- neuer Studiengang
- BSc **und** MSc
- gestartet im WS 19/20
- http://datasciencebachelor.de
- http://datasciencemaster.de
- Studiengangskoordinator: JD
- Fragen gerne an mich

Viel Erfolg in der Klausur!

Ich hoffe es hat Ihnen Spaß gemacht und Sie haben viel gelernt, was Sie in Ihrem beruflichen Werdegang weiterbringt.

Prof. Dr. Jens Dittrich

und das gesamte Vorlesungsteam:

Marcel Maltry, Tanja Bäumel, Daniel Emmel, Gideon Geier, Jakob Görgen, Luca Gretscher, Jonas Lauermann, Laura Plein, Alisa Welter, Moritz Wilhelm