ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

# Aktuelle Trends: Eine Reise von Hauptspeicherdatenbanken zu Dataspace Management Systemen

Dr. Jens Dittrich

Institut für Informationssysteme

ETH Zürich

# Zusammenfassung

- Heute
  - OLTP versus OLAP versus IR
  - Stand der Dinge in OLAP
- Morgen
  - Hardwaretrends
  - Hauptspeicherdatenbanken
- Übermorgen
  - Dataspace Management
  - iMeMex

# Heute

# OLTP: Online Transaction Processing

- Viele kleine Transaktionen
- Anfragen berühren nur kleine Bereiche der Datenbasis
- Schreib- und Lesetransaktionen
- Nebenläufigkeit und Locking
- Vermeidung von Redundanz (Normalisierung)
- Zugriff auf aktuelle Daten notwendig
- Beispiele:
  - Buchungssysteme
  - Kundenverwaltung
  - Lagerverwaltung
  - ERP
  - etc.

# OLAP: Online Analytical Processing

- Wenige grosse Anfragen
- Aufwendige Anfragebearbeitung
- Anfragen berühren große Bereiche der Datenbasis
- Nur Lesetransaktionen, keine Updates!
- Redundanz (Denormalisierung)
- Materialisierte Views (Aggregate)
- Nur Zugriff auf leicht veraltete Daten (tagesaktuell)
- Beispiele:
  - Management Information (Verkäufe pro Mitarbeiter)
  - Aktuelle Geschäftsdaten
  - Wissenschaftliche Daten (Astronomie, Bioinformatik, etc.)
  - etc.

# IR: Information Retrieval

- Suchmaschinen für Text und semistrukturierte Daten
- Viele Anfragen
- Relativ einfache Anfragebearbeitung
- Anfragen berühren große Bereiche der Datenbasis
- Nur Lesetransaktionen, keine Updates!
- Nur Zugriff auf leicht veraltete Daten (tagesaktuell)
- Beispiele:
  - Web Search
  - Enterprise Search
  - Desktop Search

# OLTP vs. OLAP vs. IR

| | OLTP | OLAP | IR |
|---|---|---|---|
| Datenzugriff | read/write | read-mostly | read-mostly |
| Datenaktualität | aktuell | veraltet | veraltet |
| Anfragepfad | über Schlüssel | über Wert | über Wert |
| **Indexierung** | änderungseffizient | **leseeffizient** | **leseeffizient** |
| Daten | (semi-) strukturiert | strukturiert | unstrukturiert |
| Anfrageoptimierung | regelbasiert | kostenbasiert | kostenbasiert |
| Parallelität | inter-query | intra-query | intra-query |
| Precision&Recall | 1 | 1 | <=1 |
| Datenvolumen | wenig bis mittel | groß | groß |

# Wichtige Konzepte von OLAP-Systemen

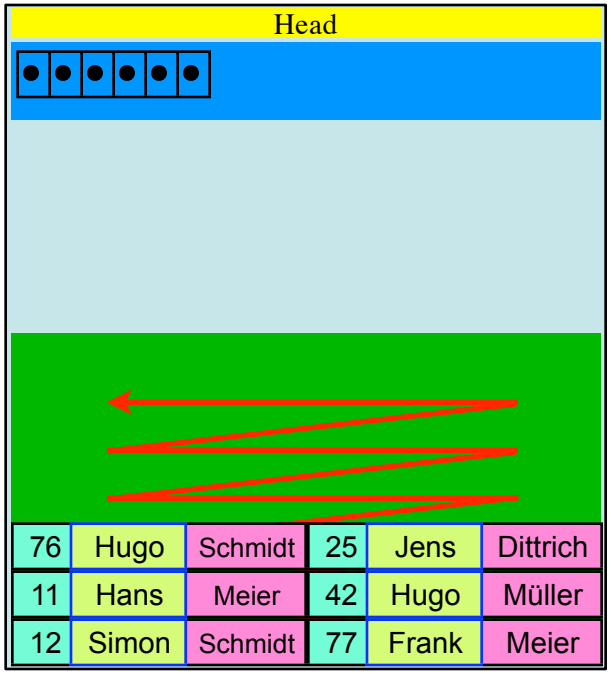
- Kostenbasierter Optimierer
- Materialisierte Sichten
- Value Bitmaps
- (Bitmap-) Joinindexe
- Parallelisierung
  - Shared Nothing: Teradata, Netezza, SAP BI Accelerator, ...
  - Shared Everything: Oracle RAC
  - Siehe Vortrag von Prof. Rahm am Dienstag
- Komprimierung (IR-style)
- Cache locality (Cache-conscious, cache-oblivious)
- Spaltenorientierung

# OLTP: Standard n-ary Storage Model (NSM)

Tabelle

| Key | fname | lname |
|---|---|---|
| 77 | Frank | Meier |
| 12 | Simon | Schmidt |
| 42 | Hugo | Müller |
| 11 | Hans | Meier |
| 25 | Jens | Dittrich |
| 76 | Hugo | Schmidt |
| | | |

Speicherseite

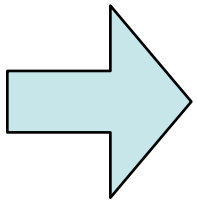



- Tupel werden zeilenweise abgelegt
- alle Attribute eines Tupels werden benachbart gespeichert

# Decomposition Storage Model (DSM)

Tabelle

| RID | Key | fname | lname |
|---|---|---|---|
| 1 | 77 | Frank | Meier |
| 2 | 12 | Simon | Schmidt |
| 3 | 42 | Hugo | Müller |
| 4 | 11 | Hans | Meier |
| 5 | 25 | Jens | Dittrich |
| 6 | 76 | Hugo | Schmidt |
| | | | |



Tabelle

| RID | Key |
|---|---|
| 1 | 77 |
| 2 | 12 |
| 3 | 42 |
| 4 | 11 |
| 5 | 25 |
| 6 | 76 |
| | |

Tabelle

| RID | fname |
|---|---|
| 1 | Frank |
| 2 | Simon |
| 3 | Hugo |
| 4 | Hans |
| 5 | Jens |
| 6 | Hugo |
| | |

Tabelle

| RID | lname |
|---|---|
| 1 | Meier |
| 2 | Schmidt |
| 3 | Müller |
| 4 | Meier |
| 5 | Dittrich |
| 6 | Schmidt |
| | |

- Teile Tabelle in Menge von Spalten auf
- Alternativen
  - zwei Attribute pro Spalte (RID und fname)
  - ein Attribut (fname), Position des Eintrag ist implizierter Schlüssel (Array-artige Repräsentation)

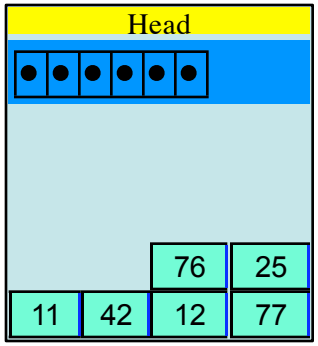
# Decomposition Storage Model (DSM)

| RID | Key |
|---|---|
| 1 | 77 |
| 2 | 12 |
| 3 | 42 |
| 4 | 11 |
| 5 | 25 |
| 6 | 76 |
| | |

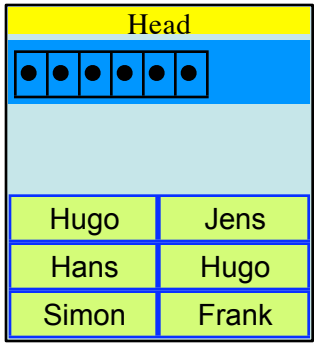
| RID | fname |
|---|---|
| 1 | Frank |
| 2 | Simon |
| 3 | Hugo |
| 4 | Hans |
| 5 | Jens |
| 6 | Hugo |
| | |

| RID | lname |
|---|---|
| 1 | Meier |
| 2 | Schmidt |
| 3 | Müller |
| 4 | Meier |
| 5 | Dittrich |
| 6 | Schmidt |
| | |

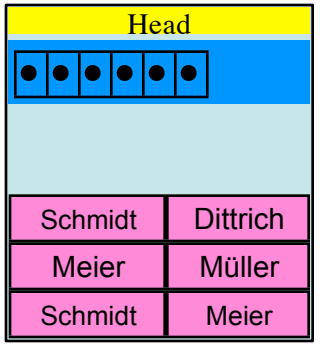
Speicherseite




Speicherseite

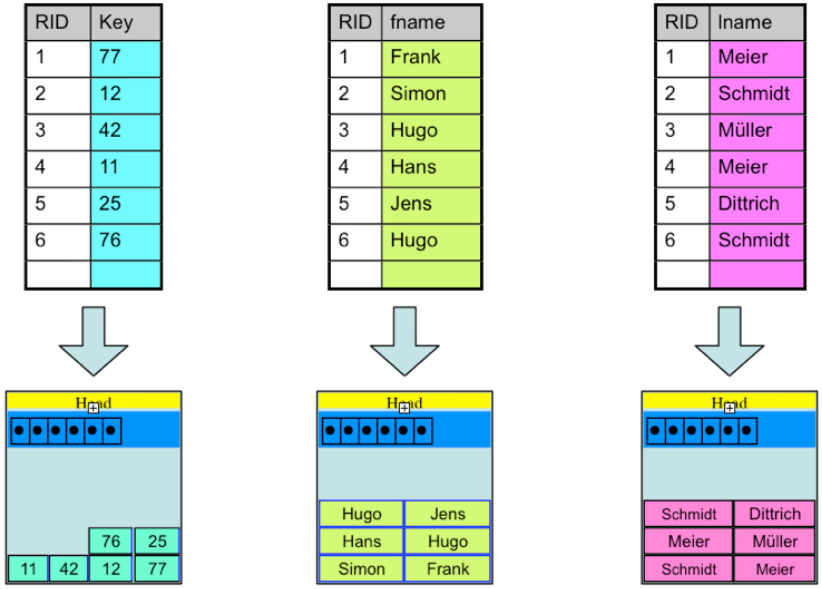



Speicherseite



# Decomposition Storage Model (DSM)

- Vorteil: effizient, wenn nur wenige Attribute benötigt werden
- Nachteil: ineffizient, wenn viele Attribute benötigt werden
- Nachteil: Tupel auf viele Seiten verteilt




- Literatur
  - Don S. Batory: On Searching Transposed Files. ACM Trans. Database Syst. 1979.
  - George P. Copeland, Setrag Khoshafian: A Decomposition Storage Model. SIGMOD 1985

# Projektionsindex

- Variante von DSM
- Idee
  - NSM wird beibehalten
  - **aber**: erzeuge zusätzliche **redundante** Projektionen auf einzelne Attribute
  - Projektionen sind äquivalent mit Spalten des DSM-Modells
- Je nach Kosten wird NSM oder Projektionsindex genutzt
- Literatur:
  Patrick E. O'Neil, Dallan Quass: Improved Query Performance with Variant Indexes. SIGMOD Conference 1997: 38-49

# Column Stores: eine Auswahl

- Sybase IQ (seit frühen 90ern)
- KDB
- Addamark (jetzt: Sensage)
- SAP BI Accelerator
- Vertica
- Bigtable (Google File System)
- zahlreiche Forschungsprototypen
  - MonetDB
  - C-Store
  - EaseDB

# Morgen

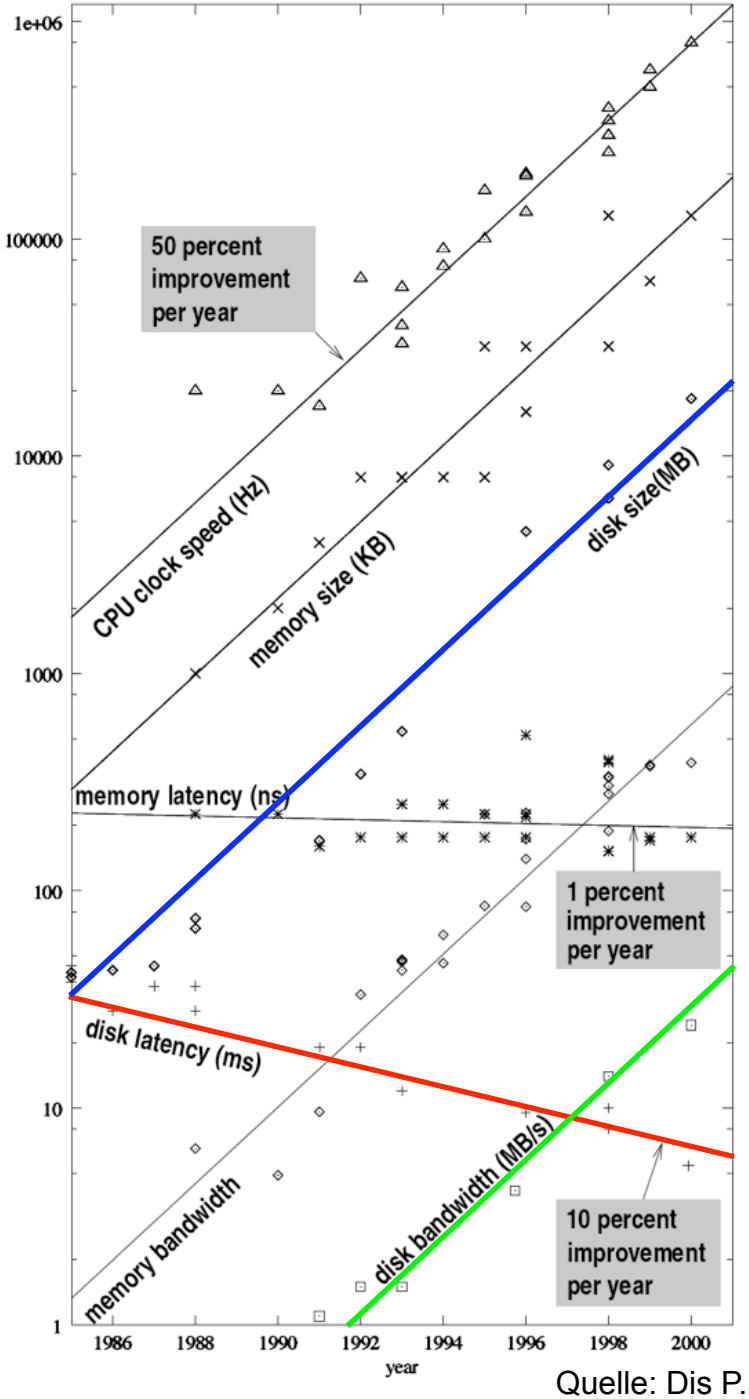
# Hardware: Festplatten

- Sehr hohe Durchsatzraten:
  ca. 190 MB/sec*
- Sehr langsam für wahlfreies Lesen, d.h. Positionieren des Schreib-/Lesekopfes:
  5-10 ms
- Wahlfreier Zugriff kann leicht zum Flaschenhals werden!

*Herstellerangaben: WD Raptor, SATA, 150 GB, 1,5 Gb/s, 16 MB Cache, 10.000 U/min, Quelle: http://www.wdc.com/de/products/products.asp?driveid=189

# Entwicklung von Festplatten

- Zugriffszeit (disk latency) verbesssert sich nur um ca. 10% pro Jahr
- Aber:
  Durchsatz (disk bandwidth) für sequentiellen Zugriff erhöht sich um ca. 50% pro Jahr!
- Ferner:
  Kapazität der Festplatten (disk size) wächst um ca. 50% pro Jahr.




Quelle: Dis P. Boncz

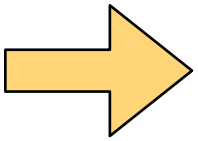
# Sequentieller vs. wahlfreier Zugriff

- **Experiment**
  Lesen von 1000 Blöcken à 8 KB

| | 1970 | 2007 | Verbesserung |
|---|---|---|---|
| wahlfrei | 48 275 ms | 6 000 ms | 8,0 |
| sequentiell | 10 315 ms | 70 ms | 147,4 |
| Verhältnis | 4,7 | 85,7 | |

**Folgerungen:**

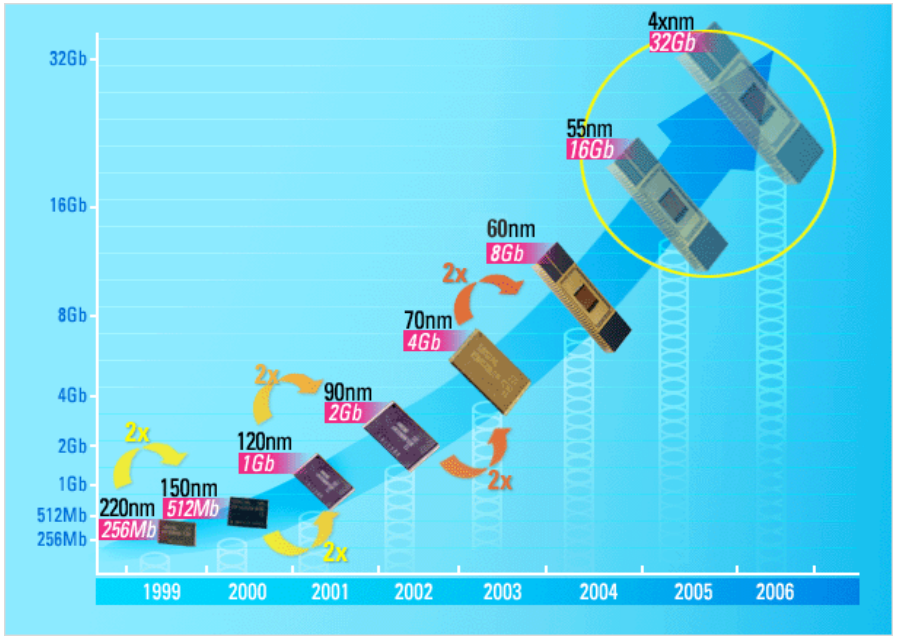
- Sollen mehr als 1/85,7= **1,1%** der Blöcke angefasst werden, lohnt sich bereits das sequentielle Lesen der **gesamten Datei!!!**
- 1970 lag dieser Wert noch bei ca. **21.3%**.



Wichtiges Designkriterium für DBMS!

# Mögliche Abhilfe: Flash Disks

- 1995 16 Mb NAND flash chips
  2005 16 Gb NAND flash
  Verdoppelt sich jedes Jahr seit 1995
- Markt getrieben von Handys, Kameras, iPods,…
  Geringe Kosten,
  ~$30/chip → ~$3/chip
- 2012 1 Tb NAND flash
  == 128 GB chip
  == 1TB oder 2TB “Festplatte”
  für ~$400
  oder 128GB Festplatte für $40
  oder 32GB Festplatte für $5




Quelle: Jim Gray, CIDR 2007

# Leistung von Flash

- Chip Lesen ~ 20 MB/s
  Schreiben ~ 10 MB/s
  N chips haben N x Bandbreite --> Flash-RAIDS!
- Zugriffszeit ~ 25 µs für Lesestart,
  ~ 100 µs zum Lesen einer “2K Seite”

  ~ 2,000 µs zum Löschen
  ~ 200 µs zum Schreiben einer “2K Seite”

- Strom ~ 1W für 8 chips und Controller

Quelle: Jim Gray, CIDR 2007

# Beispiel: SSD - Solid State Disks

- MTRON MSD-P Series mit ATA 7 Standard Interface
  - Burst Read/Write: 133 MB/sec
  - Sustained Read: 100 MB/sec
  - Sustained Write: 80 MB/sec
  - IOPS:
    - (Sequential/Random) 76,000/**16,000**
    - Access Time: weniger als 0.1 msec

ca. Faktor 100 besser als mechanische Festplatten

- Nachteil: Preis
  - ca. Preisfaktor 50 gegenüber mechanischen Platten
  - Aber: Preisfaktor wird durch Massenmarkt noch stark fallen (Faktor 10 im Jahr 2012)

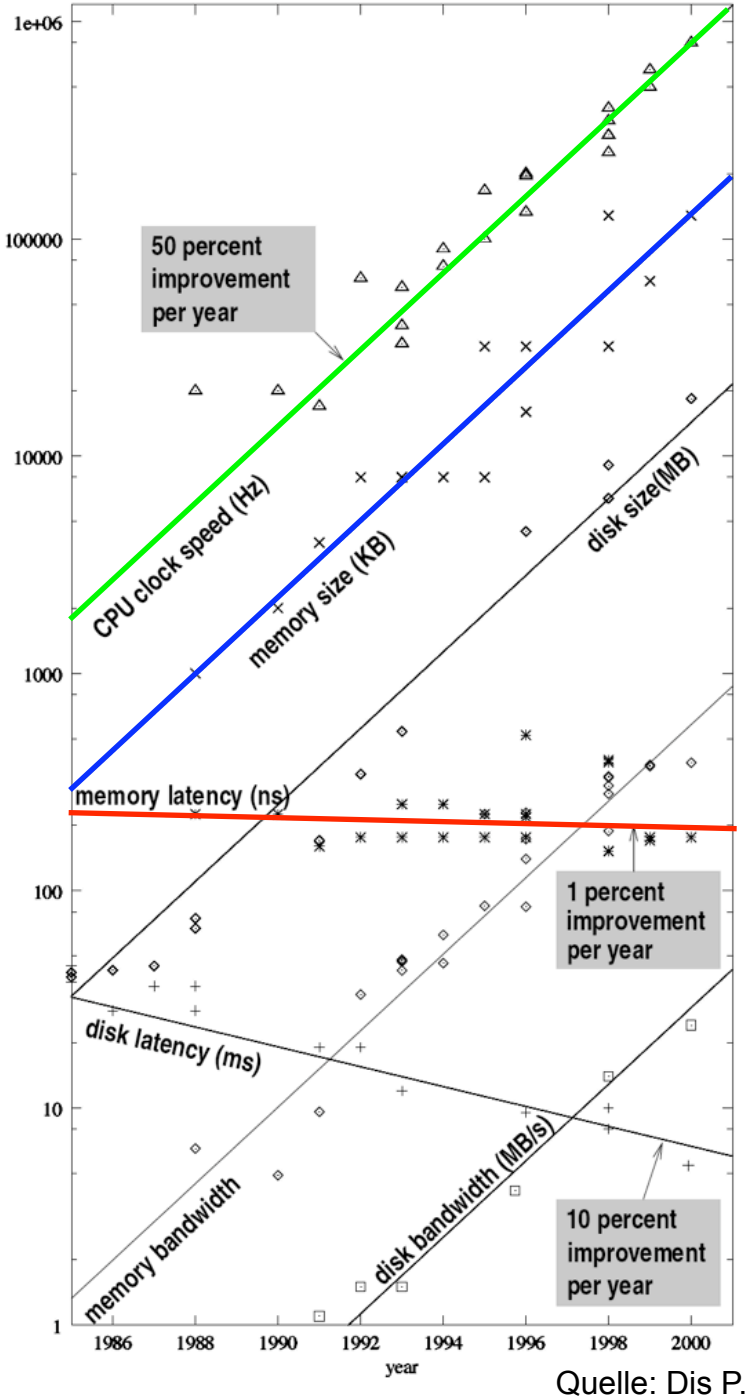
Quelle: http://www.mtron.net/eng/sub_eb11.asp

# Lösung: Hybrider Ansatz

- Kombination aus mechanischer Festplatte mit integriertem Flash-Memory
- Flash-Memory wirkt als Puffer
- Im Gegensatz zu existierenden Festplattenpuffern ist Flash persistent!
- Industrieinitiative Hybrid Storage Alliance
  - Fujitsu
  - Samsung
  - Seagate
  - Toshiba
  - Western Digital
  - Hitachi
- siehe http://www.hybridstorage.org

# Entwicklung von CPU und RAM

- Zugriffszeit (memory latency) verbesssert sich nur um ca. 1% pro Jahr
- Aber:
  CPU Taktrate (CPU clock speed) erhöht sich um ca. 50% pro Jahr!
- Ferner:
  Kapazität der Hauptspeicher (memory size) wächst um ca. 50% pro Jahr.




Quelle: Dis P. Boncz

# RAM Locality is King!

- CPU wartet meiste Zeit auf Daten aus dem Hauptspeicher (RAM)
- Flash / Festplatten sind
  100,000 …1,000,000
  Takte entfernt von der CPU
- RAM ist ~100 Takte entfernt - außer die Daten liegen im Cache.
- Um Wartezeiten zu vermeiden, sollten die Daten bei Zugriff bereits im Cache liegen: 1CPI (clock per instruction).
- Hierfür benötigt man **cache conscious** oder **cache oblivious algorithms**.
- Heutige DBMS wurden aber als diskbasierte Systeme konzipiert!

Quelle: Jim Gray, CIDR 2007

# Hauptspeicherdatenbanken

- Spezialisierte Hauptspeicherdatenbanken können erhebliche Performanzgewinne bringen
- Warum?
  - Vielen Datenbanken passen bereits komplett in den Hauptspeicher (beliebig skalierbar mittels shared nothing, Limit: $)
  - Wenn ich mit wenig $ die kompletten Daten in den HS oder auf Flash speichern kann, kann dies bereits einen Großteil meiner Performanzprobleme lösen!
- Beispiele aus der Industrie
  - Applix TM 1
  - Times Ten (2005 gekauft von Oracle)
  - SAP BI Accelerator

# 2005 Top Ten Program

## Database Size, All Environments, OLTP

| Company/Organization | Database Size (GB) | DBMS | Platform | Architecture | DBMS Vendor | System Vendor | Storage Vendor |
|---|---|---|---|---|---|---|---|
| Land Registry for England and Wales | 23,101 | DB2 | z/OS | Centralized/Cluster | IBM | IBM | IBM |
| United States Patent and Trademark Office | 16,424 | Oracle | UNIX | Centralized/SMP | Oracle | IBM | EMC |
| Elsevier | 9,616 | Oracle RAC | UNIX | Centralized/Cluster | Oracle | Sun | IBM |
| United Parcel Service | 9,284 | DB2 | z/OS | Federated/SMP | IBM | IBM | IBM |
| KTF | 8,706 | Oracle | UNIX | Centralized/SMP | Oracle | Sun | EMC |
| AIM Healthcare Services | 8,026 | SQL Server | Windows | Centralized/SMP | Microsoft | IBM | EMC |
| Verizon Communications | 7,781 | SQL Server | Windows | Centralized/SMP | Microsoft | HP | EMC |
| Anonymous | 6,800 | Sybase ASE | UNIX | Centralized/SMP | Sybase | IBM | Hitachi |
| US Bureau of Customs & Border Protection | 5,986 | CA-Datacom | z/OS | Distributed/SMP | CA | IBM | Hitachi |
| Anonymous | 5,973 | SQL Server | Windows | Centralized/SMP | Microsoft | Unisys | Hitachi |

## Database Size, All Environments, DW

| Company/Organization | Database Size (GB) | DBMS | Platform | Architecture | DBMS Vendor | System Vendor | Storage Vendor |
|---|---|---|---|---|---|---|---|
| Yahoo! | 100,386 | Oracle | UNIX | Centralized/SMP | Oracle | Fujitsu Siemens | EMC |
| AT&T | 93,876 | Daytona | UNIX | Federated/SMP | AT&T | HP | HP |
| KT IT-Group | 49,397 | DB2 | UNIX | Centralized/Cluster | IBM | IBM | Hitachi |
| AT&T | 26,713 | Daytona | UNIX | Federated/SMP | AT&T | Sun | Sun |
| LGR - Cingular Wireless | 25,203 | Oracle | UNIX | Centralized/SMP | Oracle | HP | HP |
| Amazon.com | 24,773 | Oracle RAC | Linux | Centralized/Cluster | Oracle | HP | HP |
| Anonymous | 19,654 | DB2 | UNIX | Centralized/MPP | IBM | IBM | EMC |
| UPSS | 19,467 | SQL Server | Windows | Centralized/SMP | Microsoft | Unisys | EMC |
| Amazon.com | 18,558 | Oracle RAC | Linux | Centralized/Cluster | Oracle | HP | HP |
| Nielsen Media Research | 17,685 | Sybase IQ | UNIX | Centralized/SMP | Sybase | Sun | EMC |

Quelle: wintercorp.com

# Hauptspeicherdatenbanken

- Frage: „Meine Ultra Large Database paßt niemals in den Hauptspeicher!“
- Antwort: „Muß sie auch garnicht!“
- Warum
  - nur kritische Teile der DB müssen in den Hauptspeicher
  - Analytische Anfragen müssen selten auf der gesamten DB ausgeführt werden.
  - Voraggregation reduziert Datenbestand oft erheblich!
  - Normalisierung, Komprimierung, Partitionierung, Materialisierte Views, die richtigen Index, etc.
- Machen Sie sich klar:
  „Welches Szenario soll meine DB unterstützen: OLTP, OLAP, Reporting oder IR?“ (völlig unterschiedliche Anforderungen)
- Hauptspeicherdatenbank kann ULDB erheblich von Peaks entlasten.

# Übermorgen

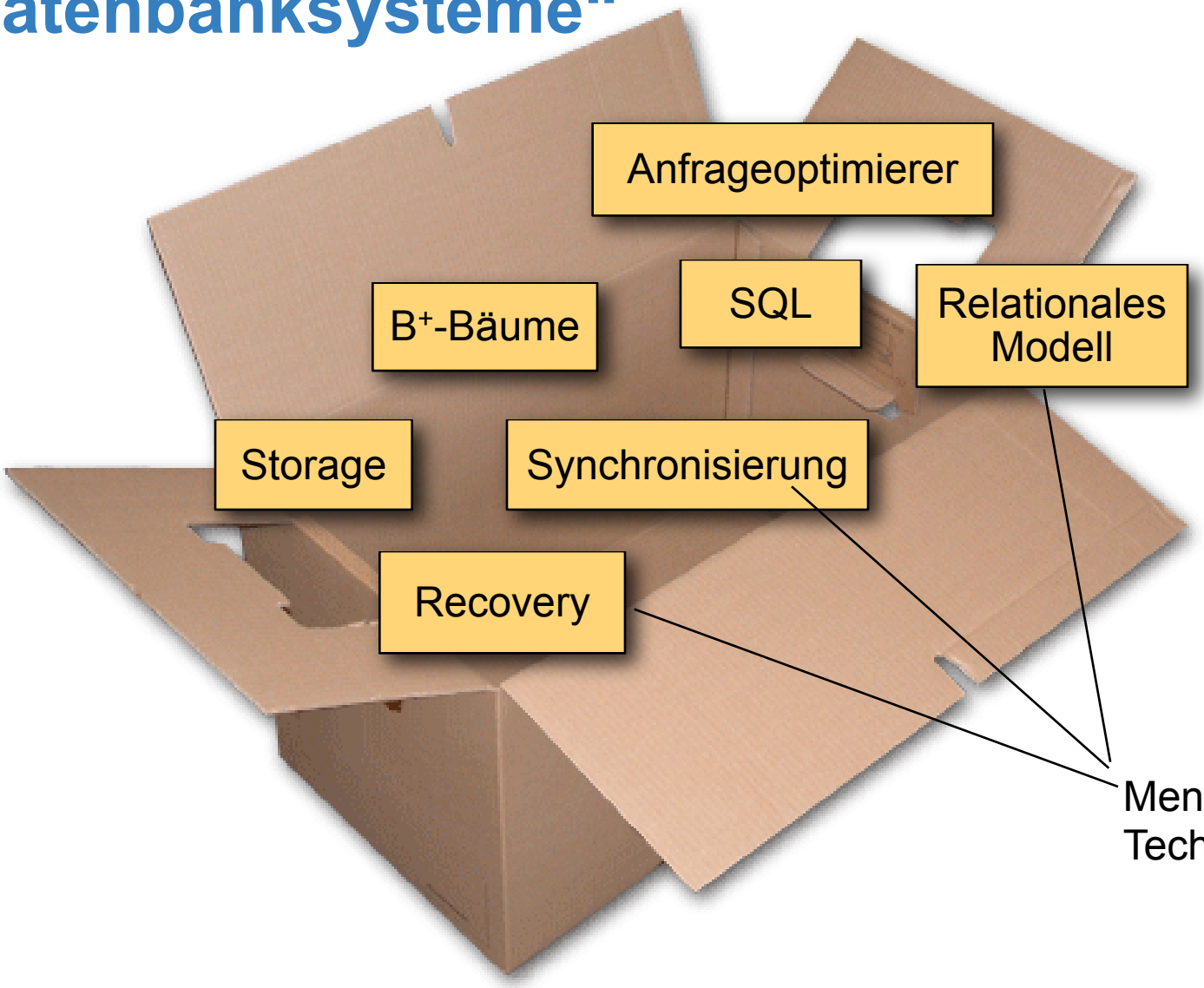
"Vorhersagen sind schwierig, insbesondere wenn sie die Zukunft betreffen."

Niels Bohr, Nobelpreis für Physik 1922

"Die beste Art die Zukunft vorherzusagen ist sie zu erfinden."

Alan Kay, ACM Turing Award 2003

# „Datenbanksysteme“



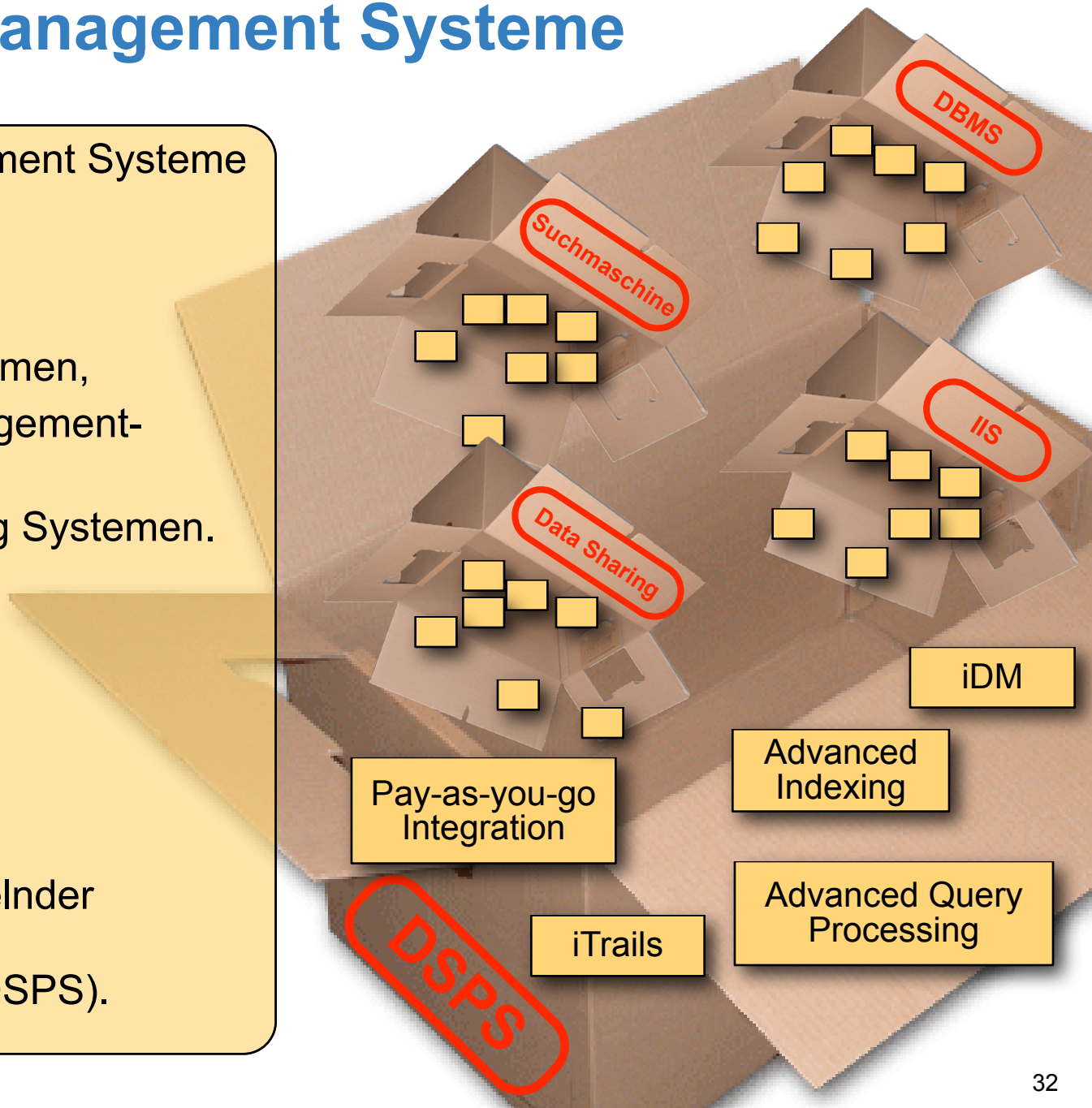
# „Datenbanksysteme“

# From Databases to Dataspaces

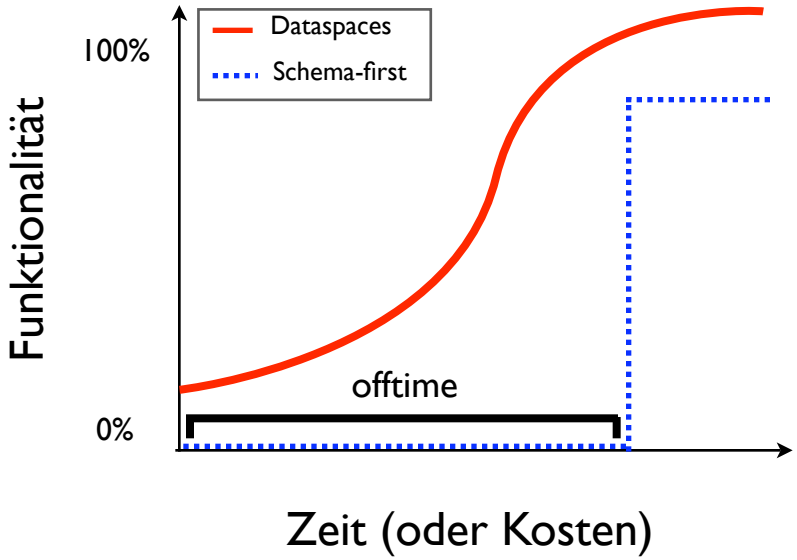
- Der Begriff „Datenbank“ ist bloß eine **Übereinkunft**, eine Abstraktion.
- Für viele Datenmanagementprobleme ist diese Abstraktion viel zu eng gefaßt.
- Beispiele:
  - Textsuche, Suchmaschinen
  - Data Streams (Eventprocessing, Pub/Sub)
  - Lose Informationsintegration mit vielen Datenquellen
- Vermutung: wir reden noch nicht über die richtige Abstraktion (die richtige Box).
- Literatur: M. Franklin, A. Halevy, D. Maier
  From Databases to Dataspaces: A New Abstraction for Information Management
  SIGMOD Record, 34(4):27–33, Dezember 2005.

# DataSpace Management Systeme

- DataSpace Management Systeme sind ein Hybrid aus:
  - Suchmaschinen,
  - Informations-integrationssystemen,
  - Datenbankmanagement-systemen,
  - und Data Sharing Systemen.
- DB + IR + IIS = Dataspace Management
- Die Summe dieser einzelnen Boxen plus neu zu entwickelnder Techniken bilden ein Dataspacesystem (DSPS).



# Wichtiges Merkmal von Dataspaces: Pay-as-you-go Information Integration

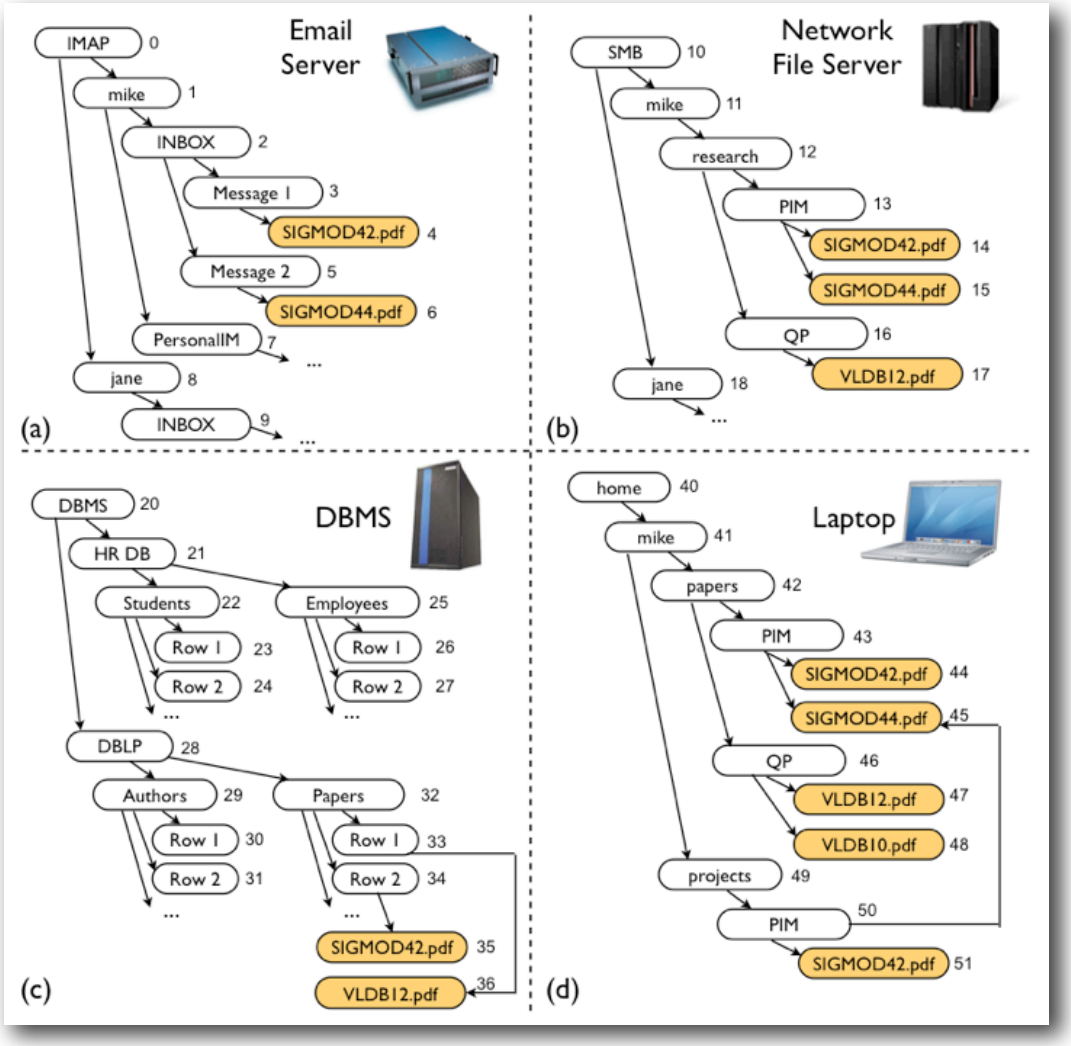



- OLAP, DWH, Enterprise Integration: benötigt komplexe Mappings und Schemata bevor Anfragen an das System gestellt werden können.
- Dataspaces: stellt einfache Dienste zur Verfügung, die mit der Zeit inkrementell verbessert werden.

# Pay-as-you-go Informationsintegration

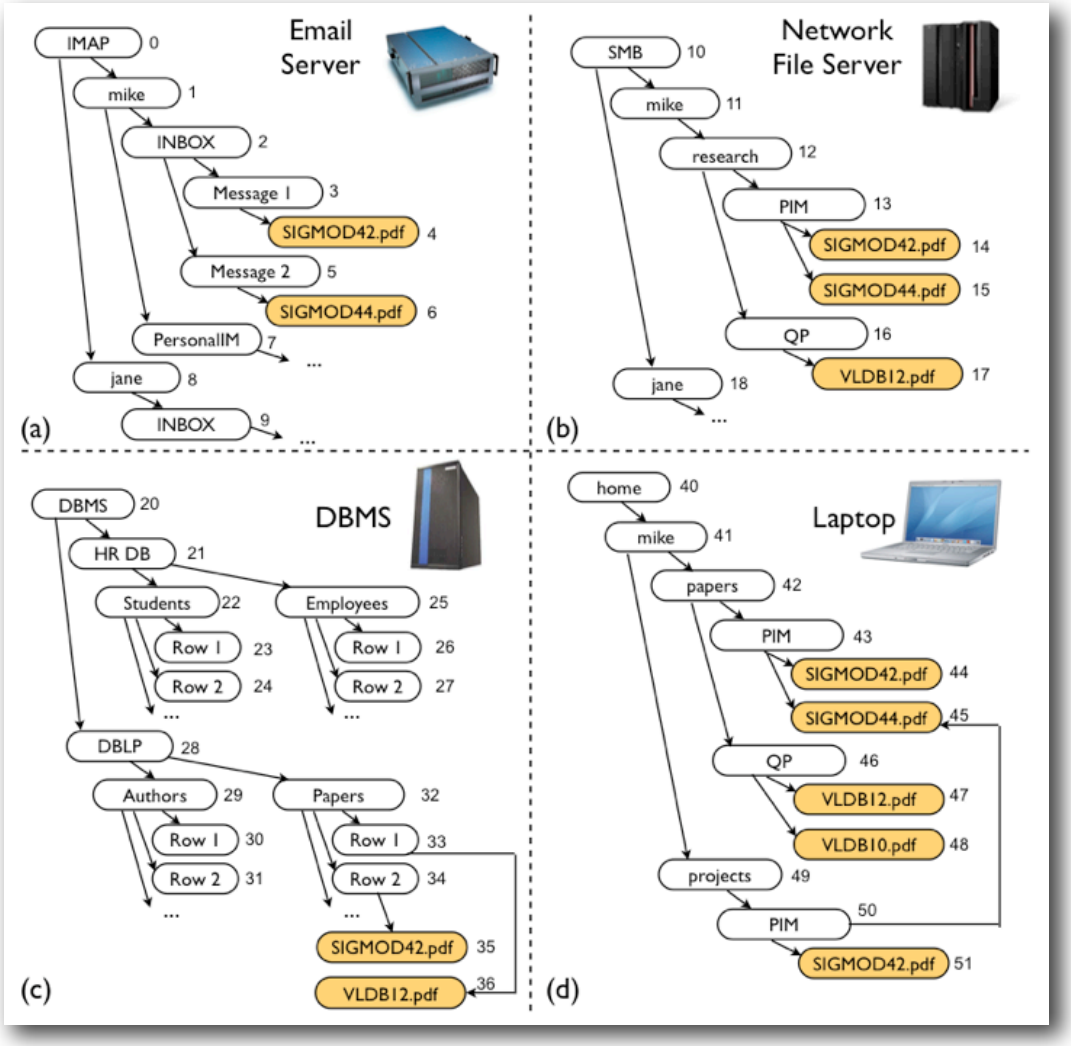
- Kernidee
  - starte mit Suchmaschine
  - keine Datenschemata notwendig
  - Daten werden einfach beim System registriert und von diesem indiziert
  - Einfache Suchanfragen von Anfang an möglich
- Ziel
  - Anfrageergebnisse sollen dieselbe Qualität haben wie Daten aus einem OLAP-Szenario mit komplexem Schema.
- Wie?
  - Benutzergetriebene Integration: **Pay-as-you-go Informationsintegration**

# Problem: Benutzergetriebene Integration (1/2)




- **benutzergetrieben = pay-as-you-go**
- Anfrage 1:
  „Zeige mir alle PDF-Dokumente, die heute oder gestern geändert oder hinzugefügt wurden.“
- Problem:
  Unterschiedliche Schemata für jede Datenquelle
  - received
  - lastmodified
  - changed
  - etc.

# Problem: Benutzergetriebene Integration (2/2)

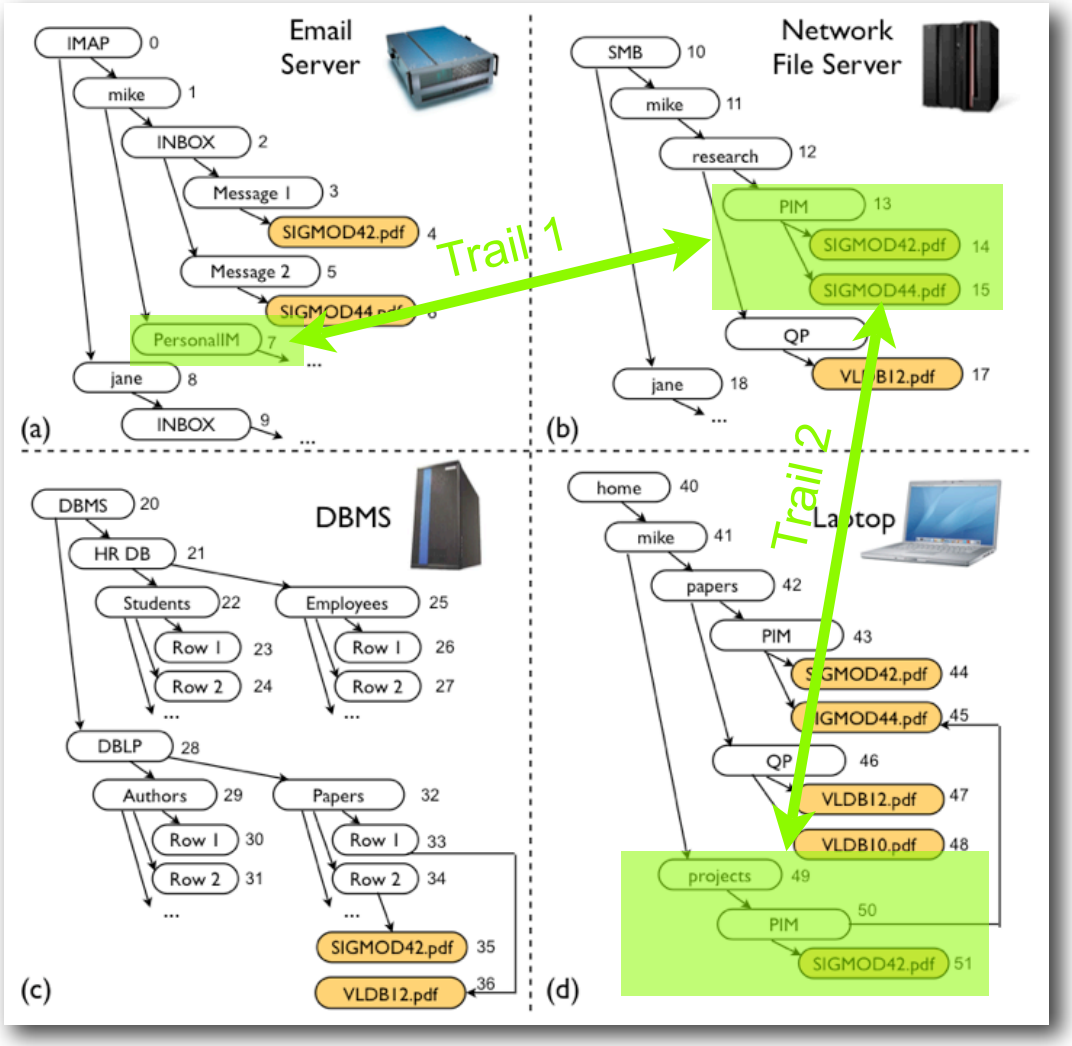



- Anfrage 2:
  „Zeige mir alle Dokumente des PIM Projektes.“
- Problem:
  Informationen sind verstreut auf verschiedene Verzeichnisse
  - //projects//PIM
  - //mike/research/PIM
  - //mike/PersonalIM
  - etc.

# Lösung: Deklaratives iTrails Framework

- Generischer Ansatz für ***Pay-as-you-go*** Informationsintegration in Dataspaces
- iTrails erlaubt leichtgewichtige Integration **ohne globales Schema**
- M. Salles, J. Dittrich, S. Karakashian, O. Girard, L. Blunschi: **iTrails: Pay-as-you-go Information Integration in Dataspaces.** to appear at VLDB 2007, Vienna, Austria.
- Kernidee:
  Benutzer/Administrator **kann zu jeder Zeit Hinweise geben** wie Daten zueinander in Beziehung stehen.
- Diese Hinweise nennen wir **Trails**.
- Drei Klassen von Trails: **semantic**, value und lineage Trails

# Lösung: Integration mit Trails



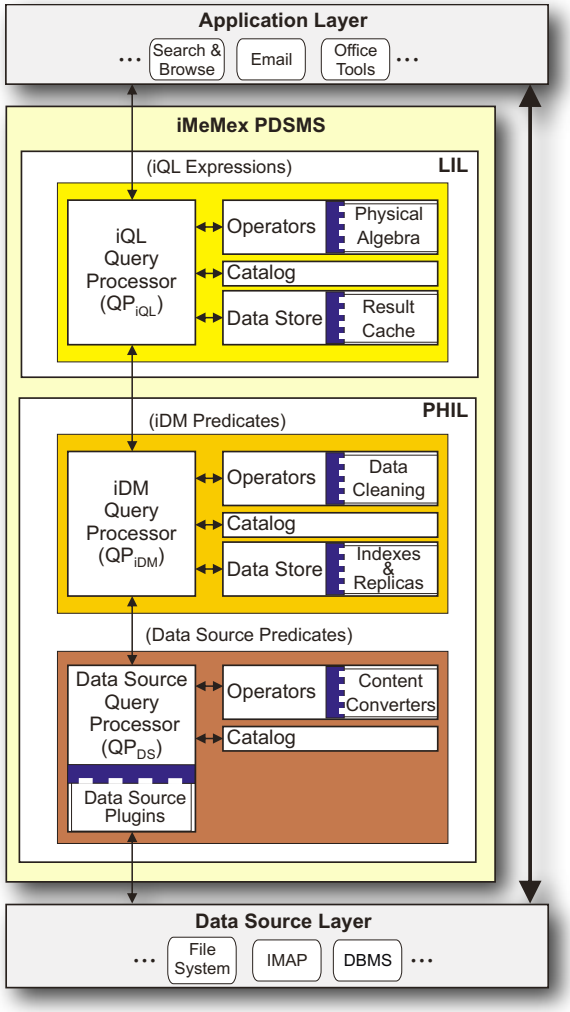

- Anfrage 2:
  „Zeige mir alle Dokumente des PIM Projektes.“
- Trails stellen Mappings zwischen Informationsfragmenten her.
  - //projects//PIM
  - //mike/research/PIM
  - //mike/PersonalIM
  - etc.

# Optionen für das Erstellen von Trails

1. Administrator legt Menge von Trails fest.
2. Diese Menge können Benutzer dann beliebig erweitern.
   (Unterstützung durch GUI möglich)
3. System kann Trails vorschlagen (Trail Mining);
   kann kombiniert werden mit Feedback-Phase
4. Trails könnten ausgetauscht werden zwischen Nutzern,
   - z.B. spezialisierte Trails für wissenschaftlichen Communities
   - Wikipedia-Plattformen
   - del.icio.us: Bookmarks (im Grunde ein Spezialfall von einem Trail)

# iMeMex DataSpace Management System

- seit Ende 2004
- Beta Version 0.44.0 open source (Apache 2.0) siehe http://www.imemex.org oder http://imemex.sourceforge.net/
- Team:
  - ich selbst (Projektleiter)
  - 2 Doktoranden
  - bisher 3 Diplomanden
  - bisher 14 Semesterarbeiten, sowie ein HiWi
- Projekt wird unterstützt vom Schweizer Nationalfond (SNF)
- Fokus momentan auf Personal Dataspaces; aber System auch für andere Szenarien anwendbar.



# iMeMex DataSpace Management System

- Veröffentlichungen zu iMeMex auf VLDB 2005, 2006, 2007; SIGIR PIM 2006; CIDR 2007; BTW 2007, ...
- Derzeitige Arbeit
  - Erweiterung des iTrails Frameworks
    - Anfrageverarbeitung
    - Indexstrukturen
  - Skalierung in den Terabytebereich
  - Parallelisierung
  - Industriepartner

# Zusammenfassung

- Aktuelle Trends: Eine Reise von Hauptspeicherdatenbanken zu Dataspace Management Systemen
- Heute
  - OLTP versus OLAP versus IR: “Know the problem“
  - Stand der Dinge in OLAP: “Know your weapons“
- Morgen
  - Hardwaretrends: “Tape is dead, disk is tape, flash is disk“
  - Hauptspeicherdatenbanken: “RAM locality is king“
- Übermorgen
  - Dataspace Management: “pay only for a search engine, get information integration on the way“
  - iMeMex: “invent the future“

# Vielen Dank für Ihre Aufmerksamkeit!

# Fragen?

Mail: jens.dittrich at inf ethz ch
Web: http://people.inf.ethz.ch/jensdi/

© 2007 für alle Folien und Grafiken: Jens Dittrich, ETH Zürich

# Backup Folien

# Speicherhierarchie

**Kapazität, Zugriffszeit**

Bytes
1-5 ns

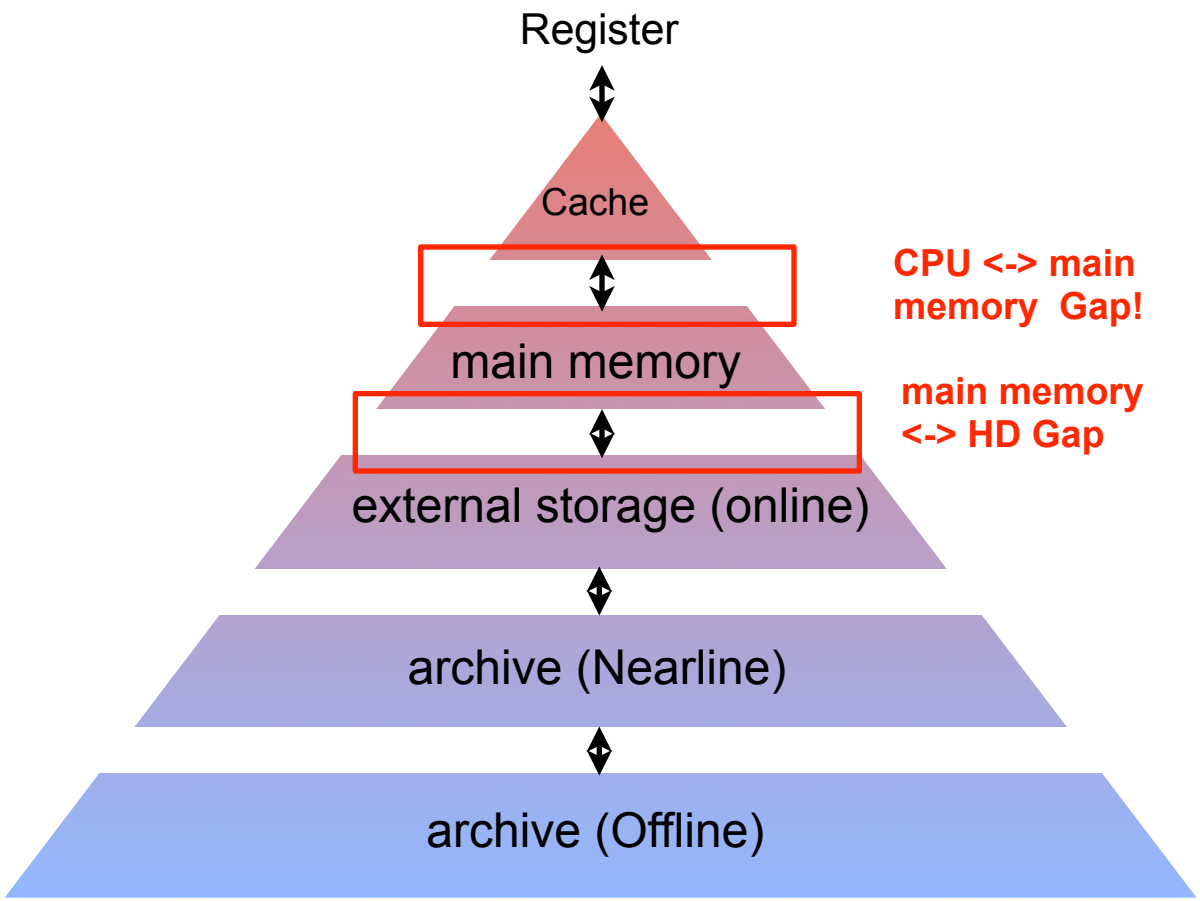
K-M Bytes
2-20 ns

M-G Bytes
50-100 ns

G Bytes
ms

T Bytes
sec

T Bytes
sec-min




**CPU <-> main memory Gap!**

**main memory <-> HD Gap**

**Kontrolle, Granularität**

Program/Compiler
1-8 Bytes

Cache-Controller
8-128 Bytes

Operating system
1-16 K Bytes

User/Operator
M Bytes (Files)

User/Operator
M Bytes (Files)

# Klassifizierung von Informationssystemen