Part I

What are Databases?

What are Databases?

What are Databases?

What are Databases?

What are Databases?

Educational Objective for Today . . .

- Motivation for using database systems

Educational Objective for Today . . .

- Motivation for using database systems
- Knowledge of basic architectures

Overview & Motivation

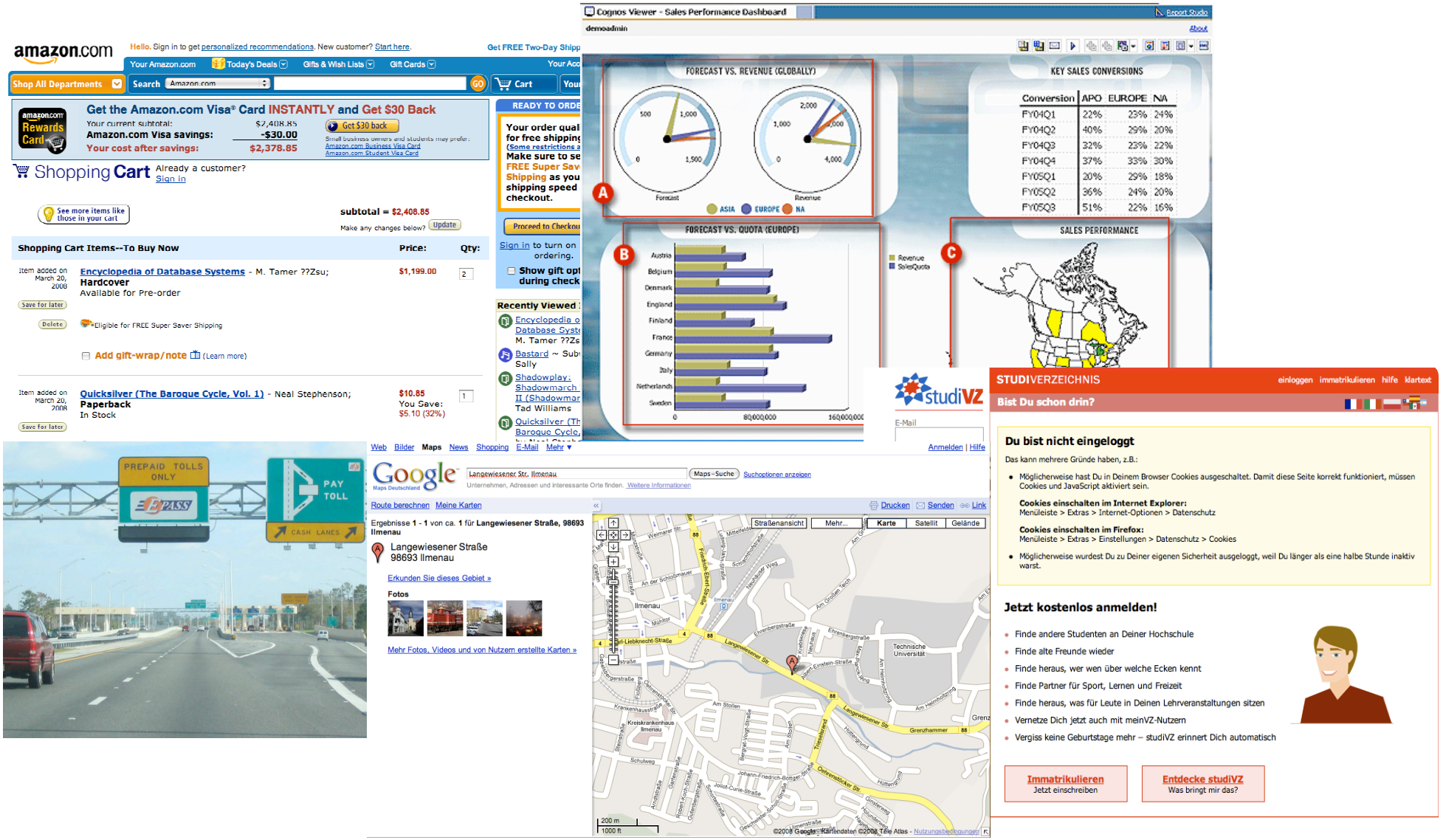
What are Databases?

- Data = logically grouped pieces of information
- Base[1]
  : the bottom or lowest part of something : the part on which something rests or is supported
  : something (such as a group of people or things) that provides support for a place, business, etc.

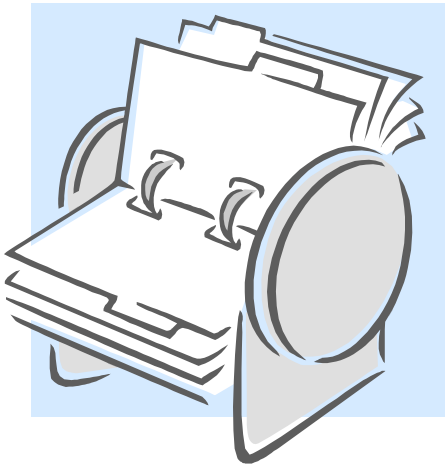
Base of a column in architecture

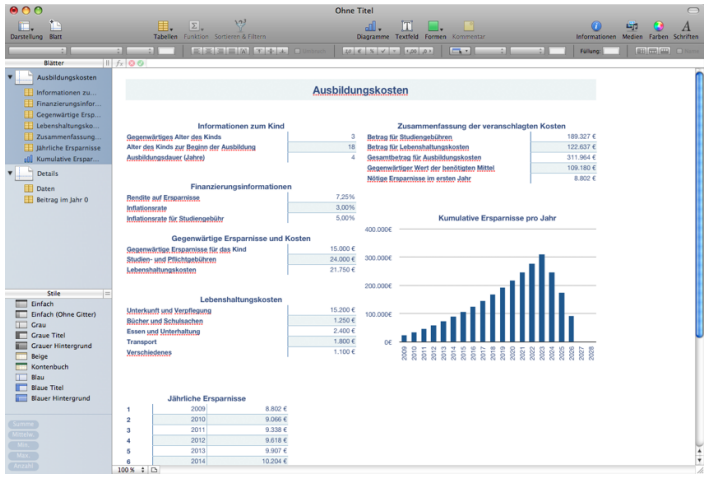
[1] http://www.merriam-webster.com/dictionary/base

Areas of Application



How is Data Managed?





Without databases

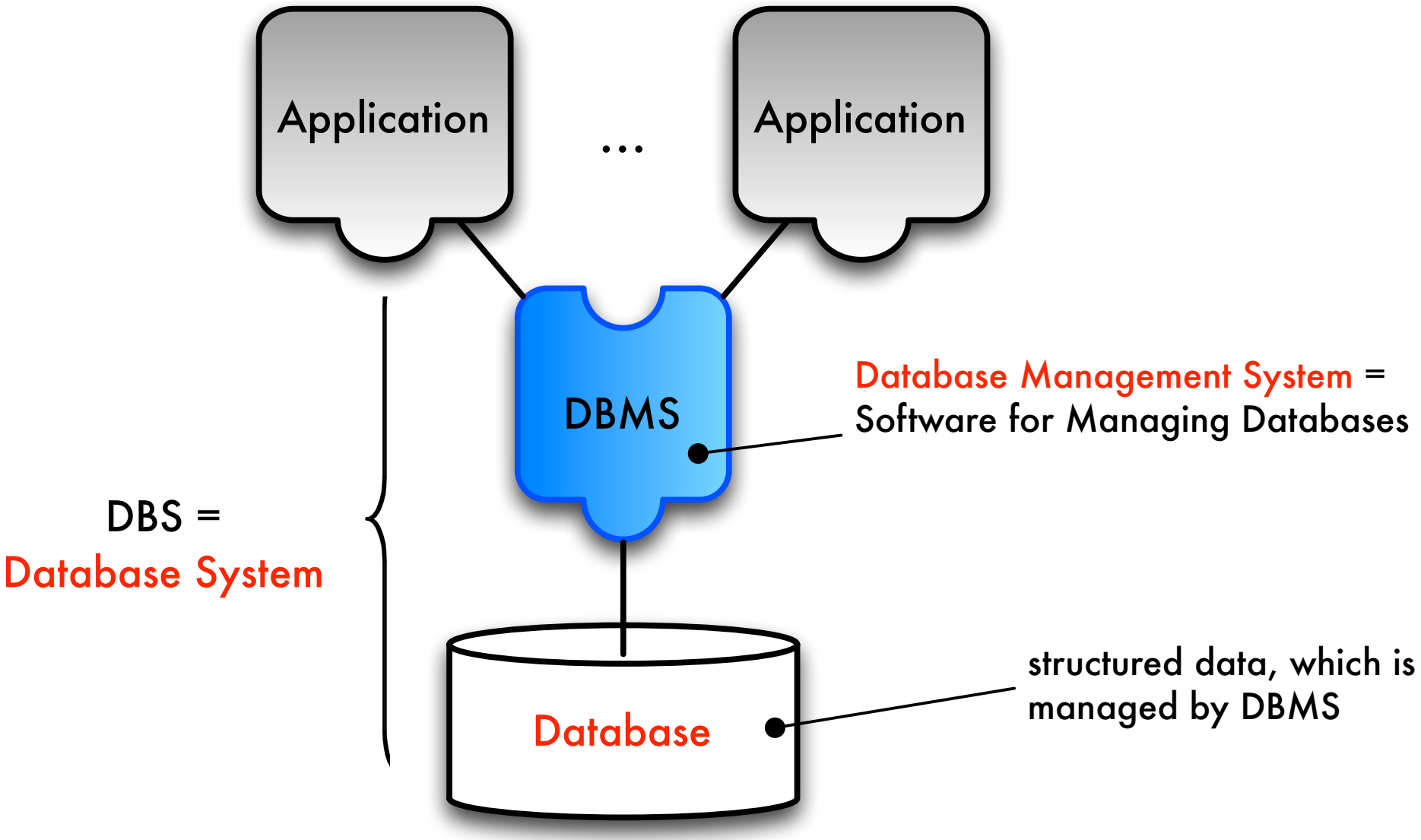
- Each application manages its own data
- Data is stored multiple times $\rightsquigarrow$ redundancy
- Problems
  - Waste of storage space
  - “Forgetting” of changes
  - No centralized, “standardized” data management

Problems of Data Redundancy

- Other software systems cannot process large amounts of data efficiently
- Many users or applications cannot access the same data in parallel without interfering with each other
- Application developers / users cannot develop / use applications without knowing
  - internal representation of data
  - storage media or computers

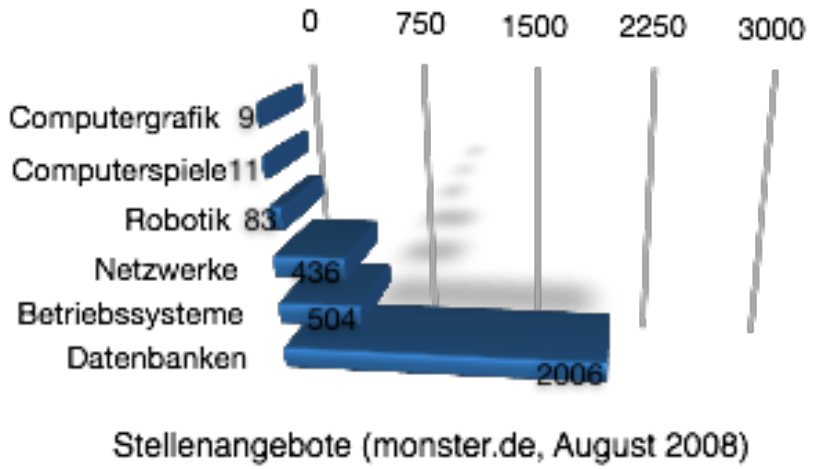
  (no data independence)
- No data security; potential loss of data

Idea: Data Integration Using Database Systems



Motivation

- Database systems are center piece of modern IT systems
- . . . ubiquitous
- Database specialists are in high demand



Stellenangebote (monster.de, August 2008)

Questions

1. How to organize (model and use) data?
2. How to store data safely and persistently?
3. How to process huge amounts of data ($\geq$ terabytes) efficiently?
4. How can many users ($\geq$ 10,000) access data concurrently?

Questions

1. How to organize (model and use) data?
2. How to store data safely and persistently?
3. How to process huge amounts of data ($\geq$ terabytes) efficiently?
4. How can many users ($\geq$ 10,000) access data concurrently?

Architectures

Principles: Codd's Nine Rules

1. **Integration**: uniform, non-redundant data management
2. **Operations**: insert, query, update, delete
3. **Catalog**: access to the database description in a data dictionary
4. **User views**: different users/applications must be able to have a different perception of the data
5. **Integrity**: ensure conformity of database contents with real world
6. **Security**: prevention of unauthorized access
7. **Transactions**: multiple DB operations handled as an atomic unit
8. **Synchronisation**: coordination of concurrent transactions
9. **Recovery**: data backup and recovery after system errors

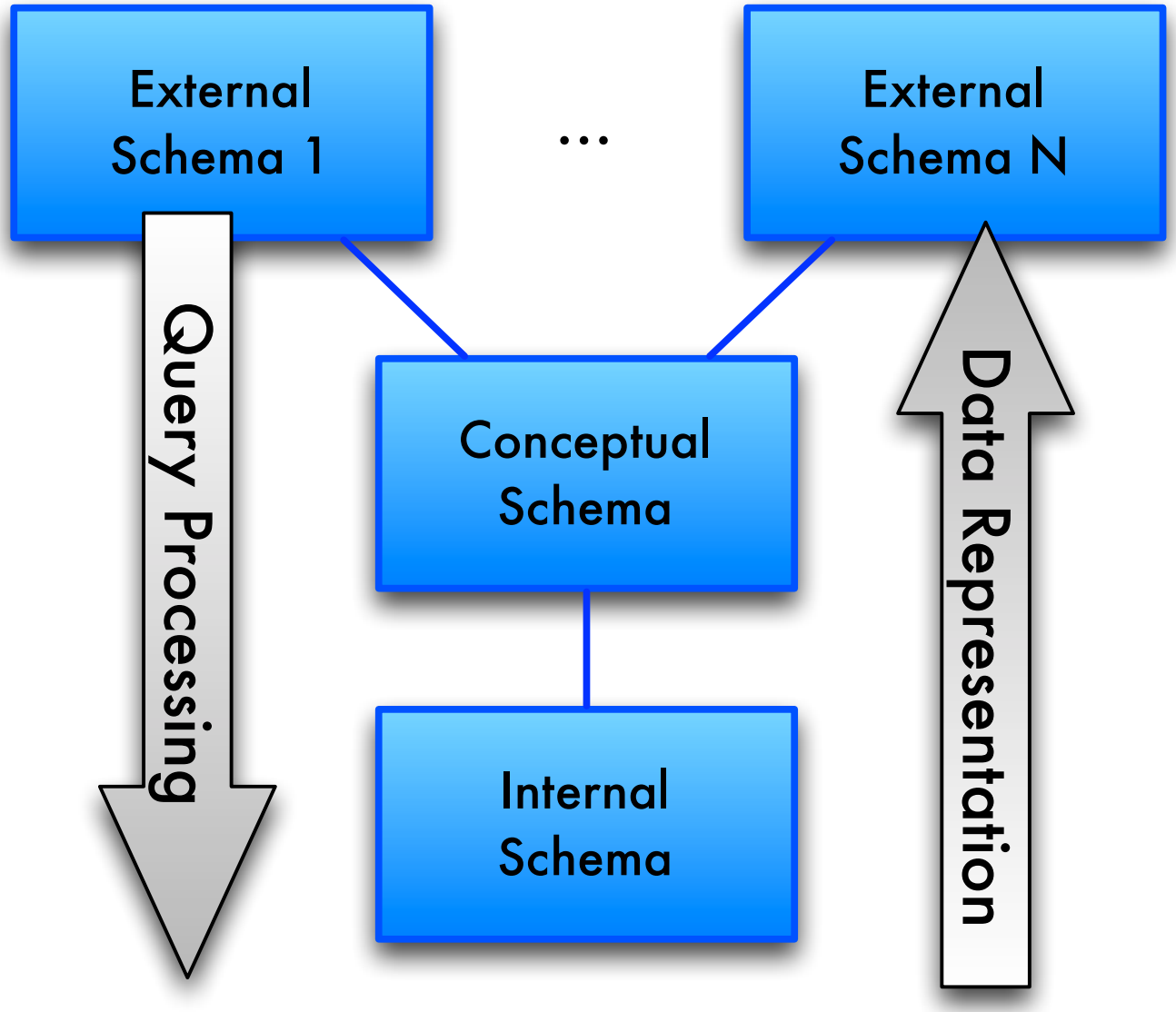
Principles: Codd's Nine Rules

1. **Integration**: uniform, non-redundant data management
2. **Operations**: insert, query, update, delete
3. **Catalog**: access to the database description in a data dictionary
4. **User views**: different users/applications must be able to have a different perception of the data
5. **Integrity**: ensure conformity of database contents with real world
6. **Security**: prevention of unauthorized access
7. **Transactions**: multiple DB operations handled as an atomic unit
8. **Synchronisation**: coordination of concurrent transactions
9. **Recovery**: data backup and recovery after system errors

Data Independence and Schemata

- Based on coarse DBMS architecture
- Decouple user and implementation view
- Goals include:
  - Separate modeling view from internal storage
  - Portability
  - Simplify tuning
  - Standardized interfaces

Schema Architecture



Schema Architecture /2

- Connection between
  - Conceptual schema (result of data definition)
  - Internal schema (definition of file structure and access paths)
  - External schemata (result of view definition)
  - Application programs (result of application programming)

Schema Architecture /3

- Separation schema — instance
  - Schema (metadata, data description)
  - Instance (user data, database state or shape)
- Database schema consists of
  - Internal, conceptual, external schemata and application programs
- Conceptual schema contains, e.g.:
  - Structure descriptions
  - Integrity descriptions
  - Authorization rules (DB accesses that a user may perform)

Data Independence /2

- Stability of user interface with respect to changes
- Physical: Changes to file structure or access paths do not influence the conceptual schema
- Logical: Changes to conceptual schema and certain external schemata do not influence other external schemata or application programs

Data Independence /3

- Potential impact of changes to the conceptual schema:
  - external schemata may be affected (changing attributes)
  - application programs may be affected (recompilation of application programs, adaptions may be necessary)
- But: necessary changes are recognized and monitored by the DBMS

Sample Application: Ride Sharing

- Offers for ride sharing
  - When? Wherefrom? Whereto? Who? Seats?
  - Contact details
  - Advance booking

CC-BY-2.0: Luo Shaoyang

Layer Architecture by Example

- Conceptual view: presentation in tables (relations)

Driver

DriverID	Name	Phone
103	Lilo Pause	01234
104	Just Vorfan	01246
105	Heiko Heizer	01756

Ride Sharing

ANr	Departure	Arrival	Time	DriverID
1014	Ilmenau	Erfurt	Friday 10:00	103
1015	Magdeburg	Halle	Friday 15:00	104
1016	Magdeburg	Leipzig	Friday 15:00	104
1021	Magdeburg	Ilmenau	Friday 14:00	105
1025	Ilmenau	Jena	Friday 10:00	103

Layer Architecture by Example /2

- External view: data in a flat relation

ANr	**Departure**	**Arrival**	**Time**	**Driver**
1014	Ilmenau	Erfurt	Friday 10:00	Lilo Pause
1015	Magdeburg	Halle	Friday 15:00	Just Vorfan
1016	Magdeburg	Leipzig	Friday 15:00	Just Vorfan
1021	Magdeburg	Ilmenau	Friday 14:00	Heiko Heizer
1025	Ilmenau	Jena	Friday 10:00	Lilo Pause

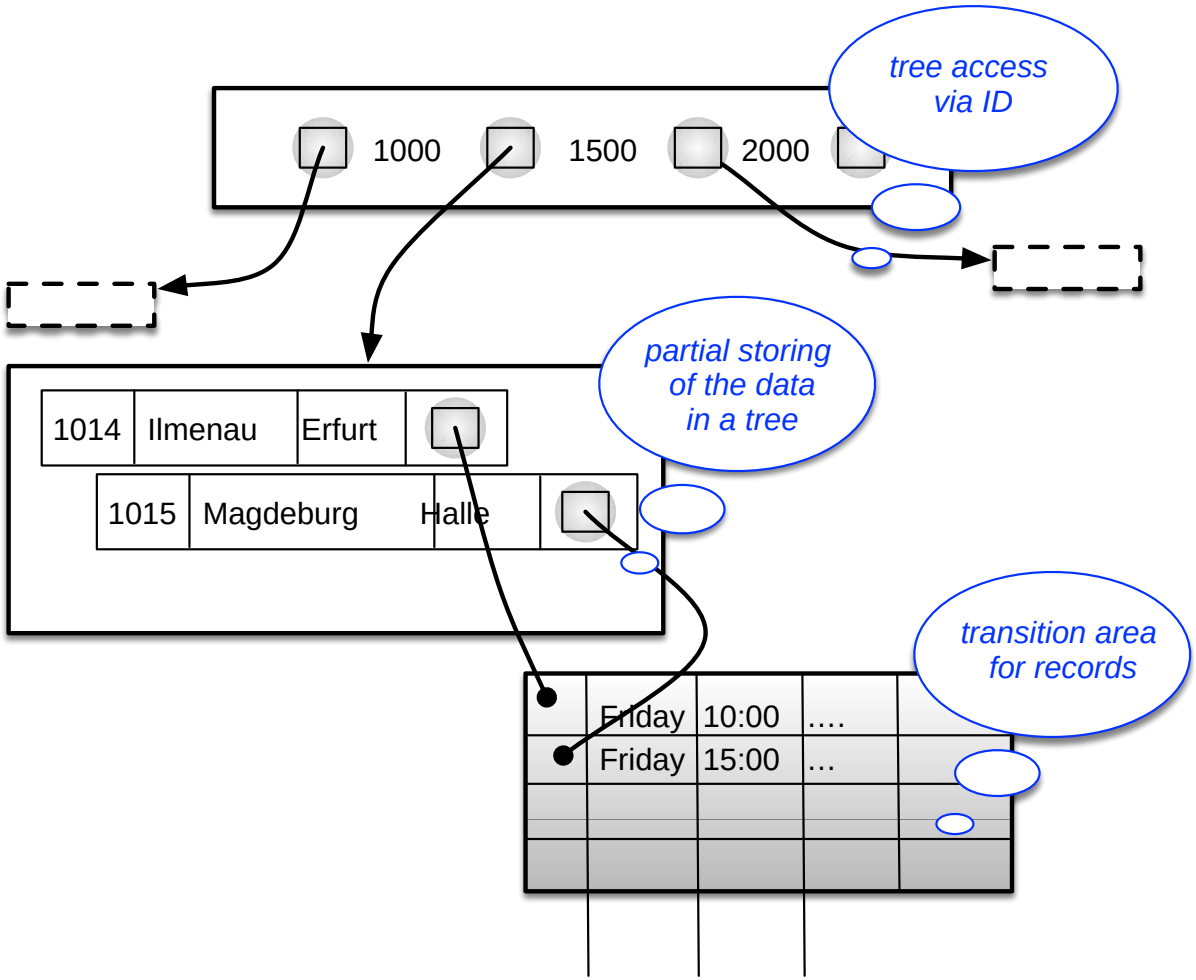
Layer Architecture by Example /3

- External view: data in a hierarchically structured relation

Driver	**Ride Share**		
	Departure	**Arrival**	**Time**
Lilo Pause	Ilmenau	Erfurt	Friday 10:00
		Jena	Friday 10:00
Just Vorfan	Magdeburg	Halle	Friday 15:00
		Leipzig	Friday 15:00
Heiko Heizer	Magdeburg	Ilmenau	Friday 14:00

Layer Architecture by Example /4

- Internal presentation



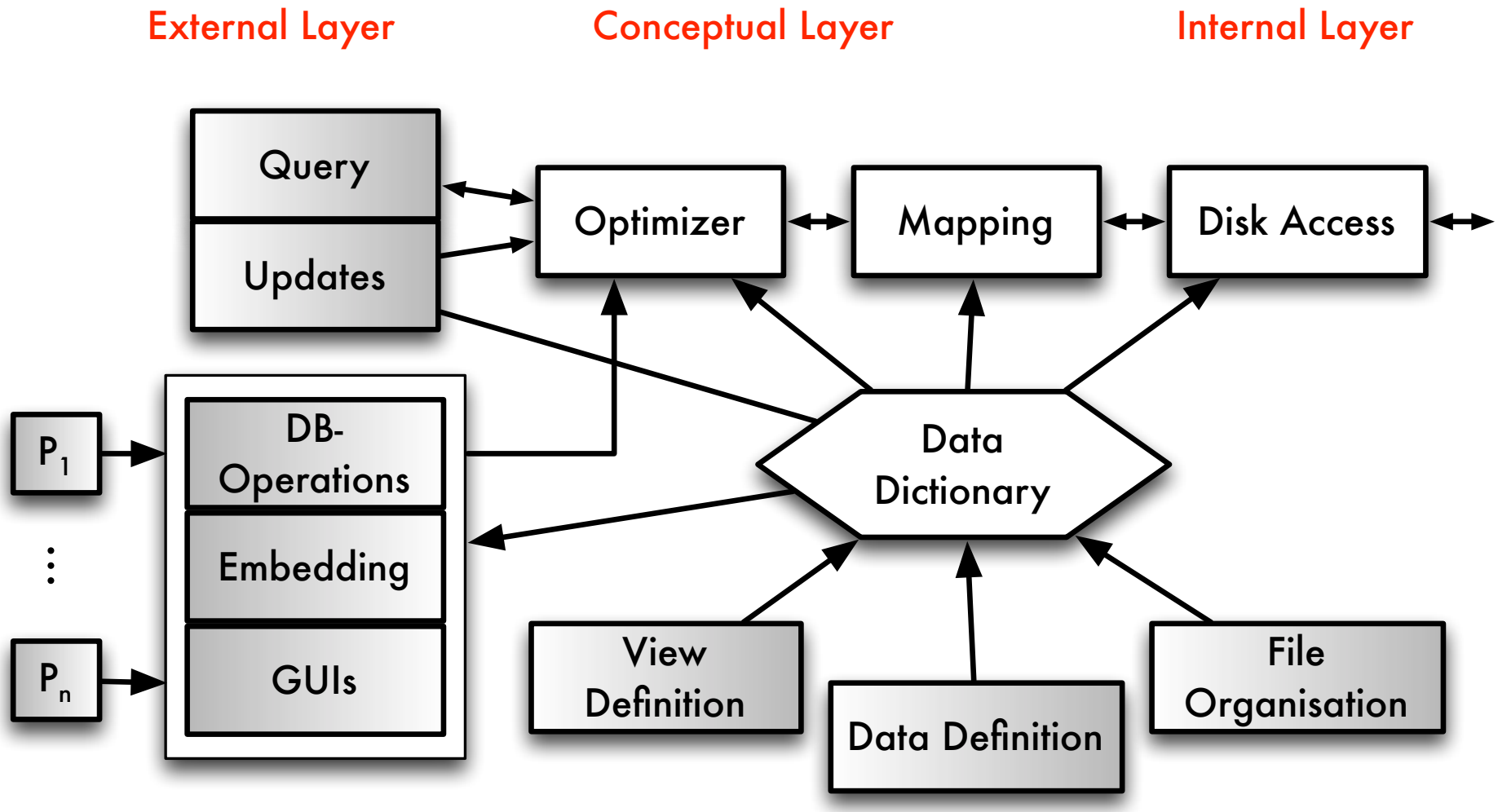
System Architectures

- Description of components of a database system
- Standardized interfaces between components
- Architecture proposals
  - ANSI-SPARC architecture
    $\rightsquigarrow$ three layer architecture
  - Five layer architecture
    $\rightsquigarrow$ describes transformation components in detail
    Lecture “Database Implementation Techniques”

ANSI-SPARC Architecture

- ANSI: American National Standards Institute
- SPARC: Standards Planning and Requirement Committee
- Proposal from 1978
- Basically refines coarse architecture
  - Internal layer / operation system refined
  - Multiple interactive and programming components
  - Interfaces named and standardized

ANSI-SPARC Architecture /2

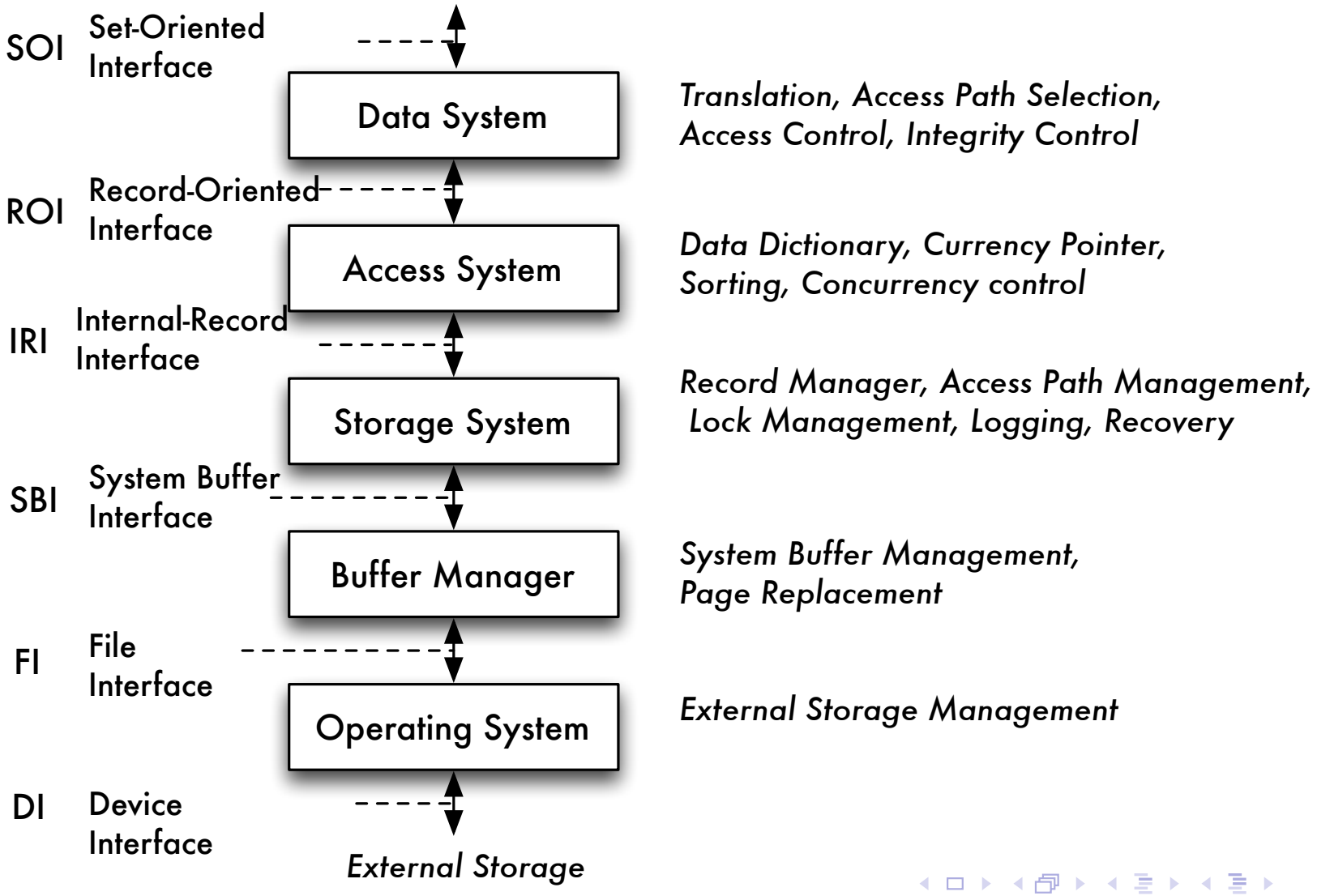


Classification of Components

- Definition components: data definition, file organization, view definition
- Programming components: DB programming with embedded DB operations
- User components: interactive application programs, query and update
- Transformation components: optimizer, analysis, disk access
- Data dictionary: contains data from definition components, provides other components with this information

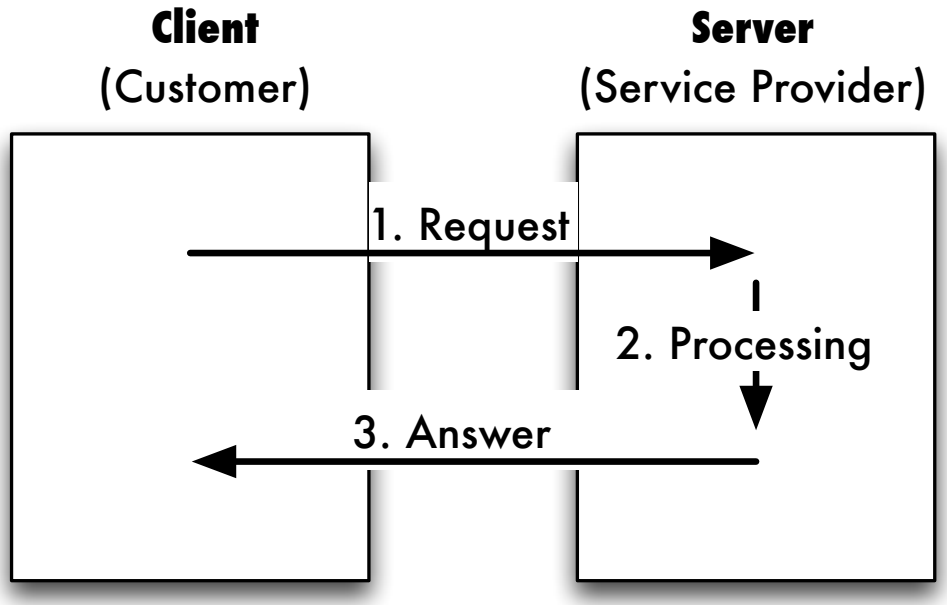
Five Layer Architecture

- Refinement of transformation steps



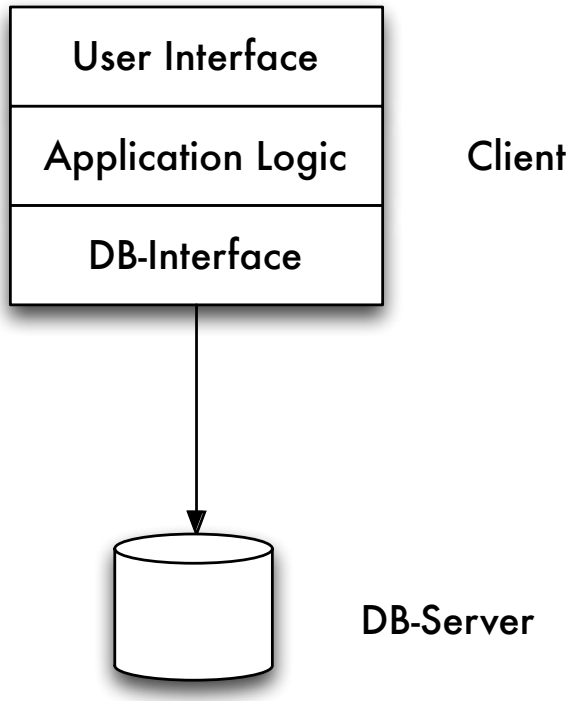
Application Architectures

- Architecture of database applications typically based on client-server model: server ≡ database system

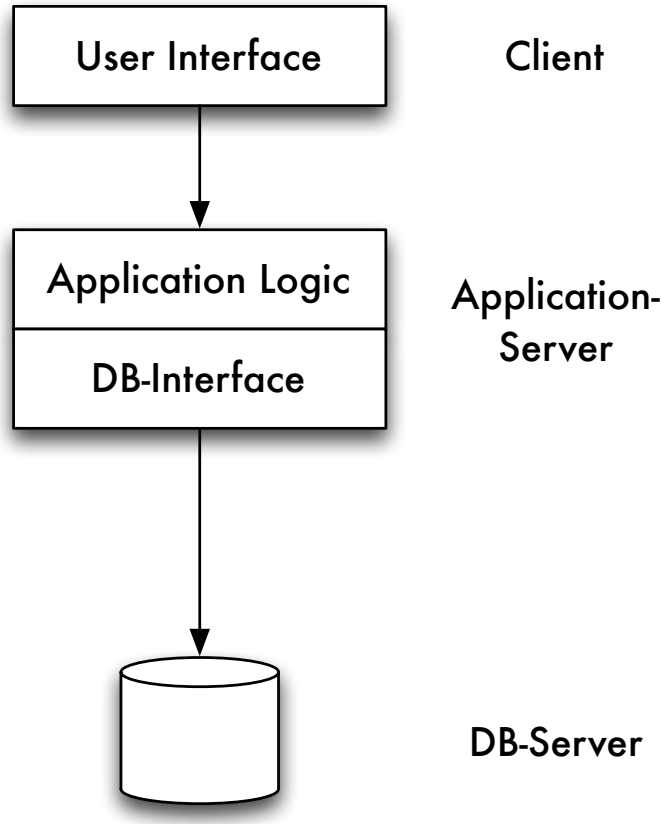


Application Architectures /2

- Separation of functionality of an application
  - Presentation and user interaction
  - Application logic (“business logic”)
  - Data management functionality (store, query, . . .).



Two Layer Architecture



Three Layer Architecture

Areas of Application

Some concrete systems

- (Object-)relational DBMS
  - Oracle11g, IBM DB2 V.10, Microsoft SQL Server 2012, SAP HANA
  - MySQL (`www.mysql.org`), PostgreSQL (`www.postgresql.org`)
- Pseudo DBMS
  - MS Access
- NoSQL systems
  - Graph database systems (InfiniteGraph, neo4j), document databases (MongoDB), key-value stores, ...

Areas of Application

- Classical areas of application:
  - Many objects (15,000 books, 300 users, 100 books borrowed per week, . . .)
  - Few object types (BOOK, USER, BORROWING)
  - For instance, book keeping systems, order tracking systems, library systems, . . .
- Current applications:
  - E-Commerce, decision supporting systems (data warehouses, OLAP), NASA's Earth Observation System (petabyte databases), data mining

Database Sizes

eBay Data Warehouse 10 PB ($\approx 10 \cdot 10^{15}$ bytes)
Teradata DBMS, 72 nodes, 10,000 users,
millions of queries/day

WalMart Data Warehouse 2,5 PB
Teradata DBMS, NCR MPP hardware;
product information (sales etc.) of 2,900 stores;
50,000 queries/week

Facebook 400 TB
x.000 MySQL server

Hadoop/Hive, 610 nodes, 15 TB/day

US Library of Congress 10-20 TB
not digitized

- PB for Petabyte is in the order of 10^{15}

History

Historical Developments: 60s

- Start of 60s: elementary files, application specific file structure (device-dependent, redundant, inconsistent)
- End of 60s: file management systems (SAM, ISAM) with service programs (sorting) (device-independent, but redundant and inconsistent)
- DBS based on hierarchical model, network model
  - Pointer structures between data
  - Weak separation of internal / conceptual layer
  - Navigational DML
  - Separation DML / programming language

Historical Developments: 70s and 80s

- 70s: database systems (device and data independence, redundancy free, consistent)
- Relational database systems
  - Data in table structures
  - 3 layer concept
  - Declarative DML
  - Separation DML / programming language

History of RDBMS

- 1970: Ted Codd (IBM) $\rightarrow$ relational model as conceptual basis of relational DBS
- 1974: System R (IBM) $\rightarrow$ first prototype of an RDBMS
  - Two modules: RDS, RSS; ca. 80,000 LOC (PL/1, PL/S, assembler), ca. 1.2 MB of code
  - Query language SEQUEL
  - First deployment 1977
- 1975: University of California at Berkeley (UCB) $\rightarrow$ Ingres
  - Query language QUEL
  - Predecessor of Postgres, Sybase, ...
- 1979: Oracle Version 2

Historical Developments: (80s and) 90s

- Knowledge base systems
  - Data in table structures
  - Strongly declarative DML, integrated database programming language
- Object-oriented database systems
  - Data in more complex object structures (separation of object and its data)
  - Declarative or navigational DML
  - Often integrated database programming language
  - Often incomplete separation of layers

Historical Developments: Today

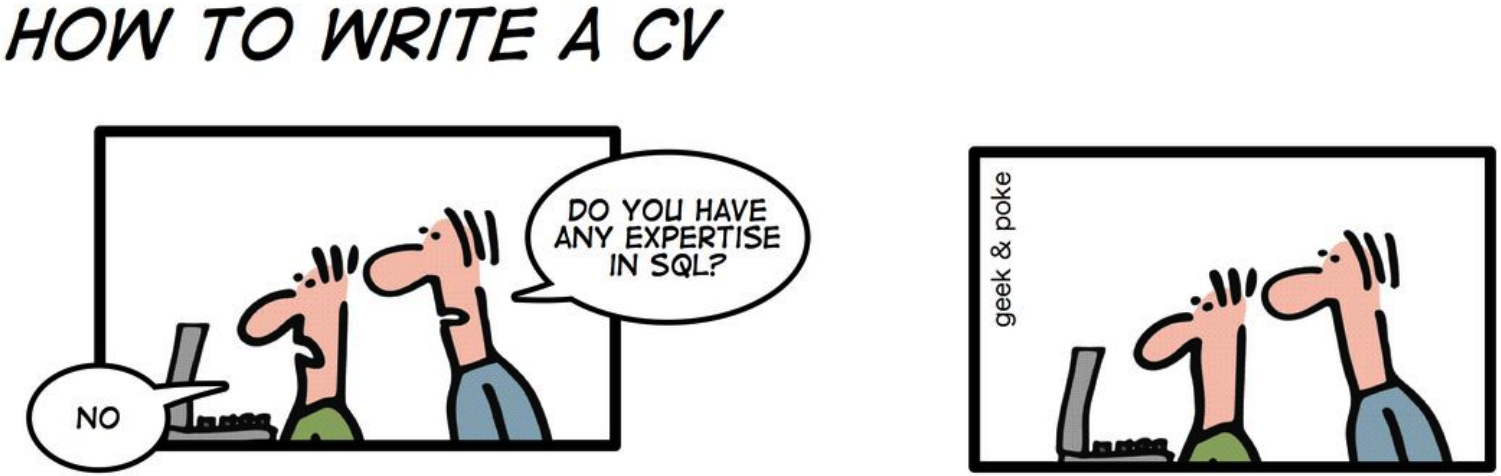
- New hardware architectures
  - Multicore processors, terabytes of main memory: in-memory database systems (e.g., SAP HANA)
- Support for specific applications
  - Cloud databases: Hosting of databases, scalable data management solutions (Amazon RDS, Microsoft Azure)
  - Data stream processing: online processing of live data, e.g., stock exchange data, sensor data, RFID data, ... (StreamBase, MS StreamInsight, IBM Infosphere Streams)
  - Big Data: Handle petabytes of data through highly scalable, parallel processing, data analysis (Hadoop, Hive, Google Spanner & F1, ...)
  - NoSQL databases ("Not only SQL"): non-relational databases, flexible schema (document-centered), "light-weight" because SQL functionality like transactions is omitted, powerful declarative query languages with joins, etc. (CouchDB, MongoDB, Cassandra, ...)

Historical Developments: NoSQL

Source: `http://geekandpoke.typepad.com/geekandpoke/2011/01/nosql.html`

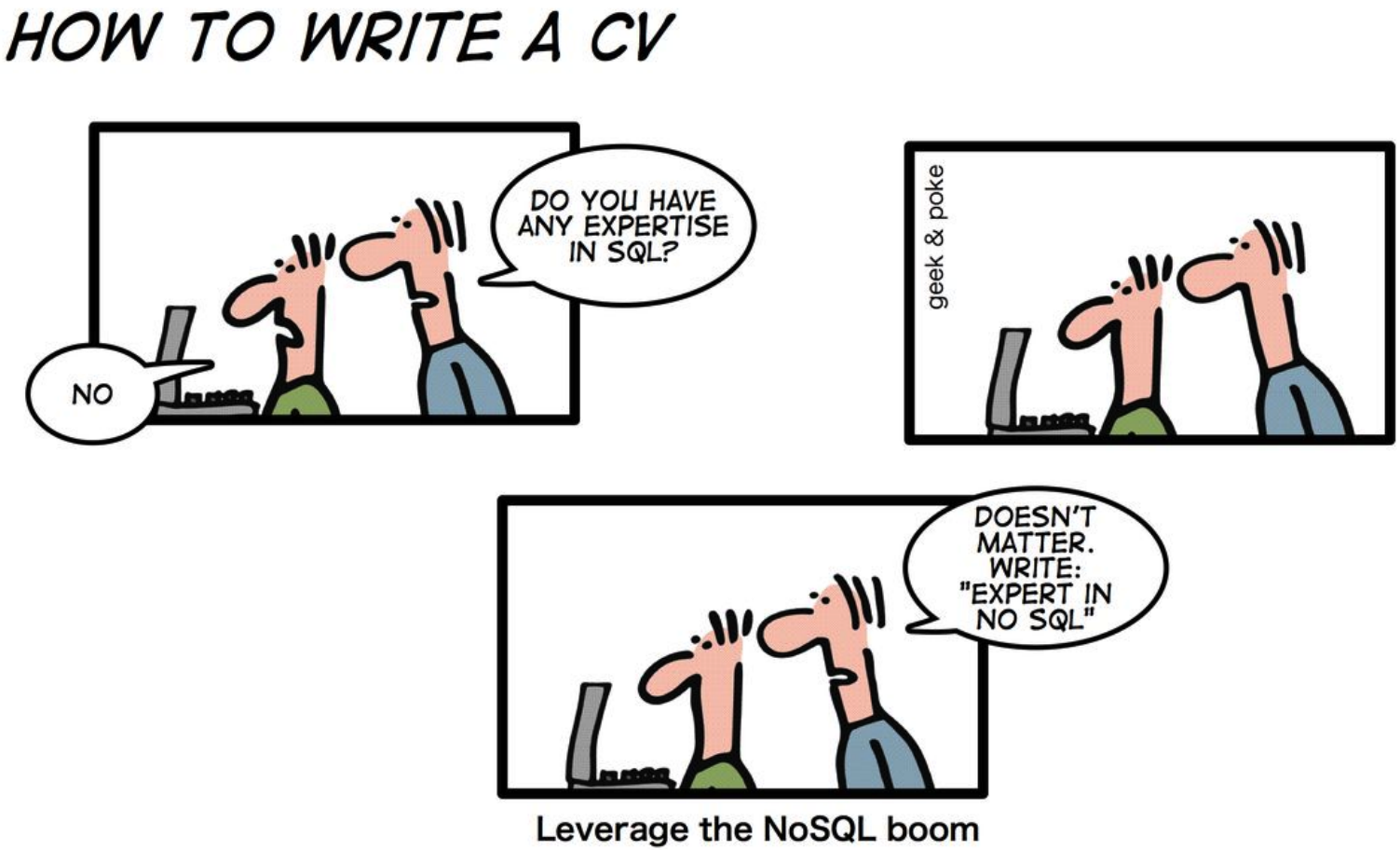
Historical Developments: NoSQL

Source: `http://geekandpoke.typepad.com/geekandpoke/2011/01/nosql.html`



Historical Developments: NoSQL

Source: `http://geekandpoke.typepad.com/geekandpoke/2011/01/nosql.html`



Trends

- User-generated content, e.g., Google:
  - Daily processing of 20 PB
  - 15 hours of video uploaded to YouTube every minute
  - Reading 20 PB would take 12 years with a 50 MB/s hard disk drive
- Linked data and data web
  - Provision, exchange and linking of structured data on the Web
  - Enables querying (with query languages like SPARQL) and further processing
  - Examples: DBpedia, GeoNames

Summary

- Motivation for using database systems
- Codd's rules
- 3 layer schema architecture & data independence
- Areas of application

Control Questions

- What is the advantage of using database systems compared to application-specific data management?

Control Questions

- What is the advantage of using database systems compared to application-specific data management?
- What does data independence mean and how is it achieved?

Control Questions

- What is the advantage of using database systems compared to application-specific data management?
- What does data independence mean and how is it achieved?
- Which are areas of application of database systems?