

Prof. DI Dr. Erich Gams

# Datenbanken

Einführung & Motivation

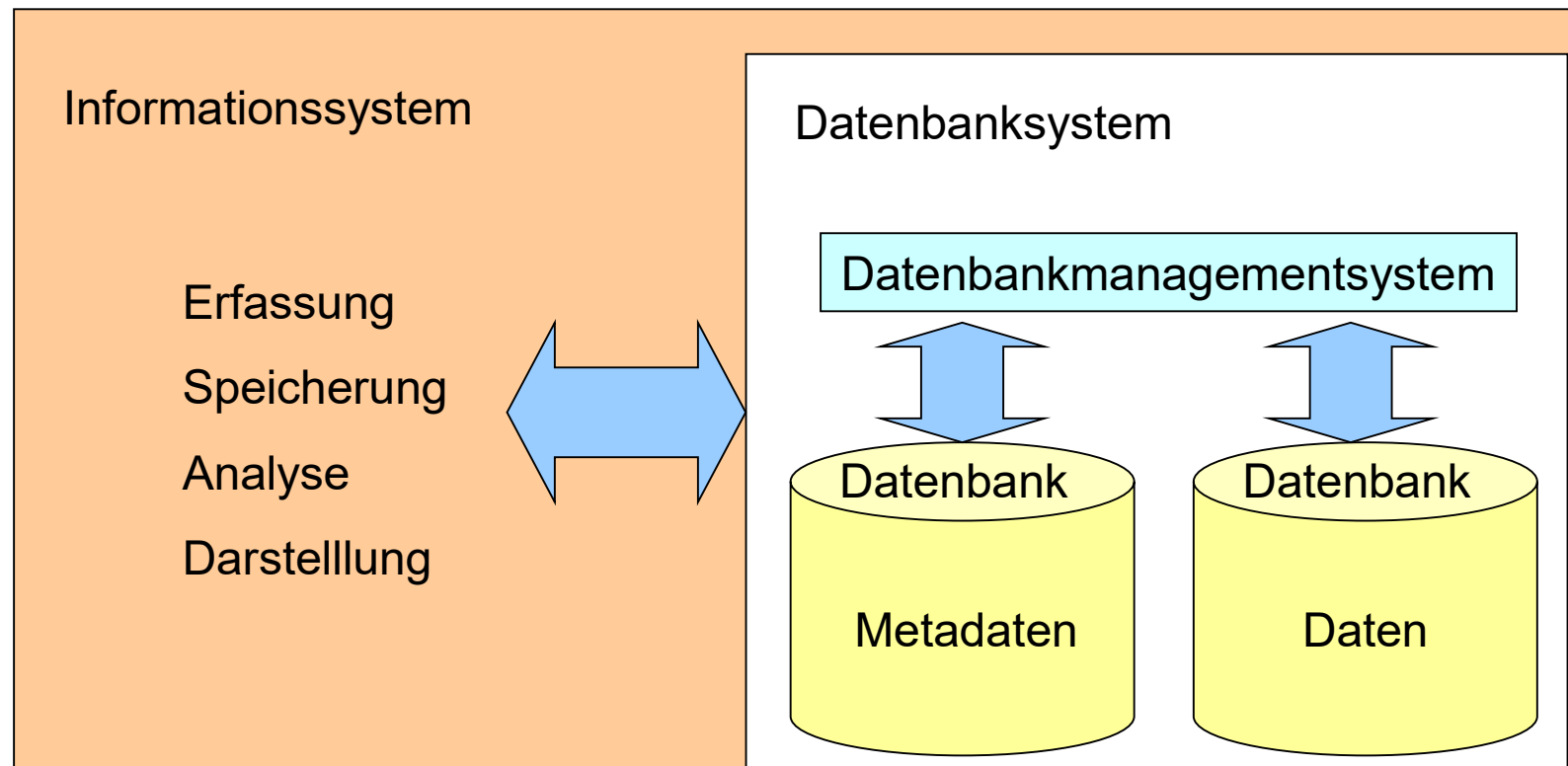
informationssysteme htl-wels

# Übersicht Was lernen wir?



- › Kurze Wiederholung
- › Vorteile DB-Ansatz
- › Geschichte & Klassifikation

# Überblick





# vom Dateisystem zur Datenbank

- › Warum speichere ich nicht in einem Dateisystem?
  - Jedes Programm verwendet eigene Dateien
  - Struktur der Dateien ist im Programm festgelegt
  
- › Nachteile
  - Abhängigkeit Programmlogik und physischer Datenstruktur (physische Datenabhängigkeit)
  - Datenstrukturänderung erfordert Programmänderung
  - Gleichzeitige Verwendung der Daten nicht möglich
  - Daten gleicher Bedeutung werden mehrfach gespeichert (Redundanz, Inkonsistenzgefahr)

# Aufgabe



- › Vor- und Nachteile einer Datenbank?

# Eigenschaften (Vorteile) von Datenbanksystemen

- › Redundanz und Konsistenz
- › Mehrbenutzersystem
- › Datensichten
- › Strukturierte und beschriebene Daten (Daten und Metadaten)
- › Integritätsverletzungen (Transaktionen)
- › Sicherheit (allgemeiner Zugriff)
- › Trennung Daten und Anwendungen
- › Persistenz
- › Weitere....?

# Nachteile DBMS

- › Wann ist kein Einsatz eines DBMS sinnvoll?

# Nachteile DBMS

- › Gefahr eines Overkills
- › Komplexität
- › Kosten
- › Qualifiziertes Personal
- › Geringe Effizienz
- › kann zu langsam sein für kritische Anwendungen
- › erhöhter Verwaltungsaufwand durch DBMS
- › Abfragesprache erfordert Wissen der relationalen Algebra

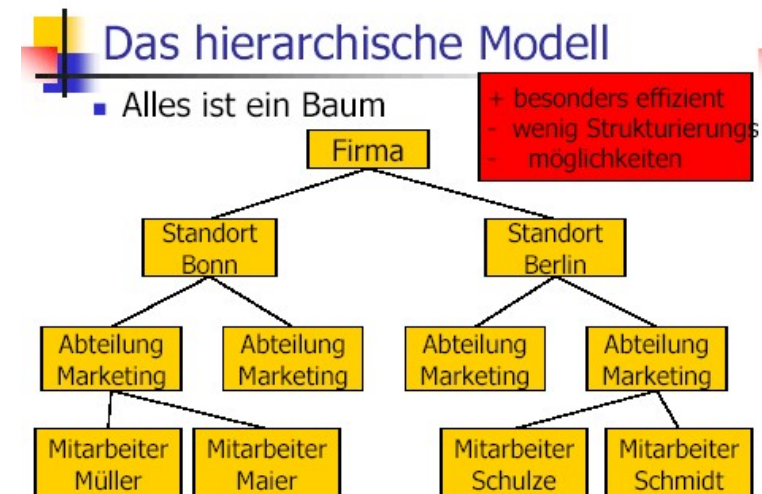


# Übersicht Arten von Datenbanken

- › Netzwerk- und hierarchische Datenbanken
- › Relationale Datenbanken
- › Objektrelationale/Objektorientierte Datenbanken
- › XML-Datenbanken
- › NoSQL Datenbanken

# Netzwerk- und hierarchische Datenbanken


- › Erste Datenbanksysteme ca. 1960-1970.
- › Solche Systeme können mit einem Stammbaum verglichen werden.
- › Es existiert **ein Wurzelknoten**, von dem sich aus wieder **mehrere Knoten verzweigen können**.



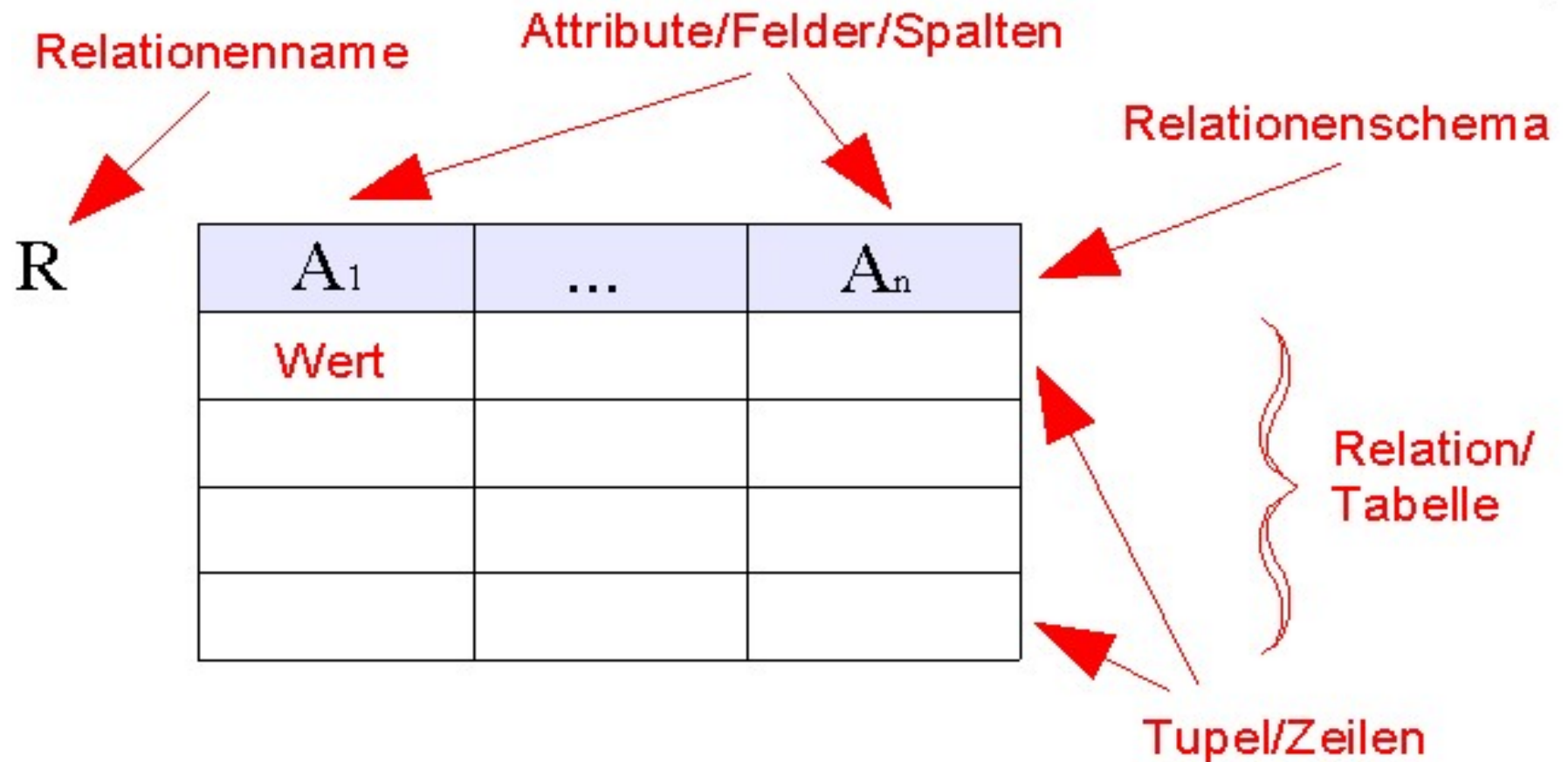
# Netzwerk- und hierarchische Datenbanken

- › Ähnlich den hierarchischen Datenbanken sind die **Netzwerkdatenbanken** strukturiert, wobei hier aber ein Kindknoten **mehrere Elternknoten** haben kann.
- › Probleme:
  - N:M Beziehung schwer abbildbar
  - Struktur rasch unübersichtlich
- › Seit den 1990er Jahren wird das Netzwerkdatenbankmodell vom relationalen Datenbankmodell mehr und mehr verdrängt.
- › Eine Renaissance erlebt die **hierarchische Datenspeicherung mit XML**.













# Relationale Datenbanken

- › „Alles ist eine Tabelle“  Relationenalgebra
- › **Daten in Tabellen** (Relationen) gespeichert, die weitgehend unabhängig voneinander sind.
- › Eine **Tabelle** ist ein zweidimensionales Gebilde aus **Spalten** (columns) und **Reihen** (rows).
- › Die **Spalten** stellen **Attribute** dar, in den Reihen werden jeweils die Werte eingetragen.
- › Die Tabellen haben lediglich **durch Schlüssel Beziehungen** zueinander.

















# Relationale Datenbanken



# Relationale DBMS Verbreitung

Rang			DBMS	Datenbankmodell
Okt 2017	Sep 2017	Okt 2016		
1.	1.	1.	Oracle 	Relational DBMS
2.	2.	2.	MySQL 	Relational DBMS
3.	3.	3.	Microsoft SQL Server 	Relational DBMS
4.	4.	4.	PostgreSQL 	Relational DBMS
5.	5.	5.	DB2 	Relational DBMS
6.	6.	6.	Microsoft Access	Relational DBMS
7.	7.	7.	SQLite 	Relational DBMS
8.	8.	8.	Teradata	Relational DBMS
9.	9.	9.	SAP Adaptive Server	Relational DBMS
10.	10.	10.	FileMaker	Relational DBMS
11.	11.	 13.	MariaDB 	Relational DBMS
12.	12.	 11.	Hive 	Relational DBMS
13.	13.	 12.	SAP HANA 	Relational DBMS
14.	14.	14.	Informix	Relational DBMS
15.	15.	15.	Vertica 	Relational DBMS

# DBMS Verbreitung allgemein


Rang			DBMS	Datenbankmodell	Punkte		
Sep 2019	Aug 2019	Sep 2018			Sep 2019	Aug 2019	Sep 2018
1.	1.	1.	Oracle 	Relational, Multi-Model 	1346,66	+7,18	+37,54
2.	2.	2.	MySQL 	Relational, Multi-Model 	1279,07	+25,39	+98,60
3.	3.	3.	Microsoft SQL Server 	Relational, Multi-Model 	1085,06	-8,12	+33,78
4.	4.	4.	PostgreSQL 	Relational, Multi-Model 	482,25	+0,91	+75,82
5.	5.	5.	MongoDB 	Document	410,06	+5,50	+51,27
6.	6.	6.	IBM Db2 	Relational, Multi-Model 	171,56	-1,39	-9,50
7.	7.	7.	Elasticsearch 	Suchmaschine, Multi-Model 	149,27	+0,19	+6,67
8.	8.	8.	Redis 	Key-value, Multi-Model 	141,90	-2,18	+0,96
9.	9.	9.	Microsoft Access	Relational	132,71	-2,63	-0,69
10.	10.	10.	Cassandra 	Wide column	123,40	-1,81	+3,85

# Objektorientierte Datenbanken



- › Objektorientierte Systeme fassen **strukturelle** und **verhaltensmäßige Komponenten** in einem **Objektyp** zusammen und gelten als die nächste Generation von Datenbanksystemen.
- › Eine Lücke wird geschlossen
  - ➡ Anwendung in einer objektorientierten Programmiersprache
  - ➡ ein klassisches relationales Datenbanksystem
- › Bsp: db4o, ObjectStore




# Nachteile OO-Datenbanken

- › kein Einzelzugriff auf die Attribute aller Objekte
- › Wenig Unterstützung von Multiuser-Anwendungen
- › Kein einheitlicher Standard
- › Anfragen:
  - Zugriffspfade zu Objekten über mehrere Pfadarten (bspw. Vererbung und Assoziation)
  - Führt bei Schreiboperationen in der Sperrverwaltung zu einer exponentiellen Komplexität und Performanceproblemen.
- ›  Geringe Verbreitung

# Objektrelationale Datenbanken

- › Bindeglied zwischen klassischen relationalen Datenbanken und Objektdatenbanken.
- › Entität  Objekt
- › Entitätstyp  Klasse
- › Nutzen die **Vorteile von objektorientierten Datenbanken** hinsichtlich der Speicherung + Abfragesprache **SQL**
- › **Bsp: Cache**

# XML-Datenbanken

- › Daten werden im XML-Format gespeichert.
- › 2 Kategorien:
  - *XML-enabled*:  
Herkömmliche Datenbanksysteme (z. B. Relationale Datenbanksysteme, ...) die ein Mapping auf oder ins XML-Format erlauben.
  - *Native XML-Datenbanksysteme*:  
Diese Systeme speichern die Information, ähnlich wie bei der Speicherung von XML-Dokumenten im Dateisystem, direkt als XML-Dokumente ab.
- › Umwandlung in XML entfällt
- › Große Datenbasis  schlechte Performance

# Literatur

- › <http://www.gitta.info/>
- › [Kemper et al.] Alfons Kemper, André Eickler: **Datenbanksysteme** Eine Einführung, 7., aktualisierte und erweiterte Auflage 2009. ISBN 978-3-486-59018-0
- › [Vossen] Gottfried Vossen, **Datenmodelle, Datenbanksprachen und Datenbankmanagementsysteme**, 5., überarbeitete und erweiterte Auflage 2008. ISBN 978-3-486-27574-2
- › [Steiner] Rene Steiner, **Grundkurs relationale Datenbanken**, 6.Auflage, Verlag Vieweg
- › [Ramez et al.] Ramez, Shamkant, **Grundlagen von Datenbanksystemen**, Auflage: 3. aktualisierte Auflage, ISBN: 978-3-8689-4012-1, Verlag Pearson

# CLIL Task: characteristics



- › 1) Describe the characteristics of different kind of databases in your own words
- › 2 ) Find out about the pros and cons of different types of databases and find some example products.
  - Network or hierachical databases
  - Relational databases
  - Object-oriented databases
  - XML-databases
  - NoSQL databases
- › 3) References (Don't forget to quote/cite!)



## What's next?

**Bill Karwin**, author of  
"SQL Antipatterns:  
Avoiding the Pitfalls of  
Database  
Programming"

3<sup>rd</sup> August 2017



## What's next?

A **semantic data model** merges relational database concepts with knowledge representation concepts from the field of AI.

Every data element has a value, but also has associations with other attributes. The idea has been around since the 1970's but (like AI) it's evolving slowly. These systems can become very expensive.



## What's next?

A **data lake** concept is a repository of data, and each set of data is left in its original format.

Structured, relational data  
Semi-structured data like CSV, logs, JSON, XML

Unstructured data like emails or other files

Binary data like images, audio, video





## What's next?

The **data lake** stores all types of data and gives you some way of querying across the whole collection. Systems like Apache Hadoop and S3 could be considered to be data lakes. It's not likely that data lakes are optimized for any particular query pattern, but rely on more brute-force strategies. As computing horsepower and parallelism becomes cheaper, no one will care.

Auf los geht's los :-)

