

Prof. Dr. Th. Härder
AG Datenbanken und Informationssysteme
Zi. 36/330, Tel.: 0631-205-4030
E-Mail: haerder@informatik.uni-kl.de
<http://www.dbis.informatik.uni-kl.de/>

Datenbankanwendung

Wintersemester 2001/2002

Universität Kaiserslautern
Fachbereich Informatik
Postfach 3049
67653 Kaiserslautern

Vorlesung:

Ort: 48 - 208

Zeit: Mo., 10.00 - 11.30 Uhr

und

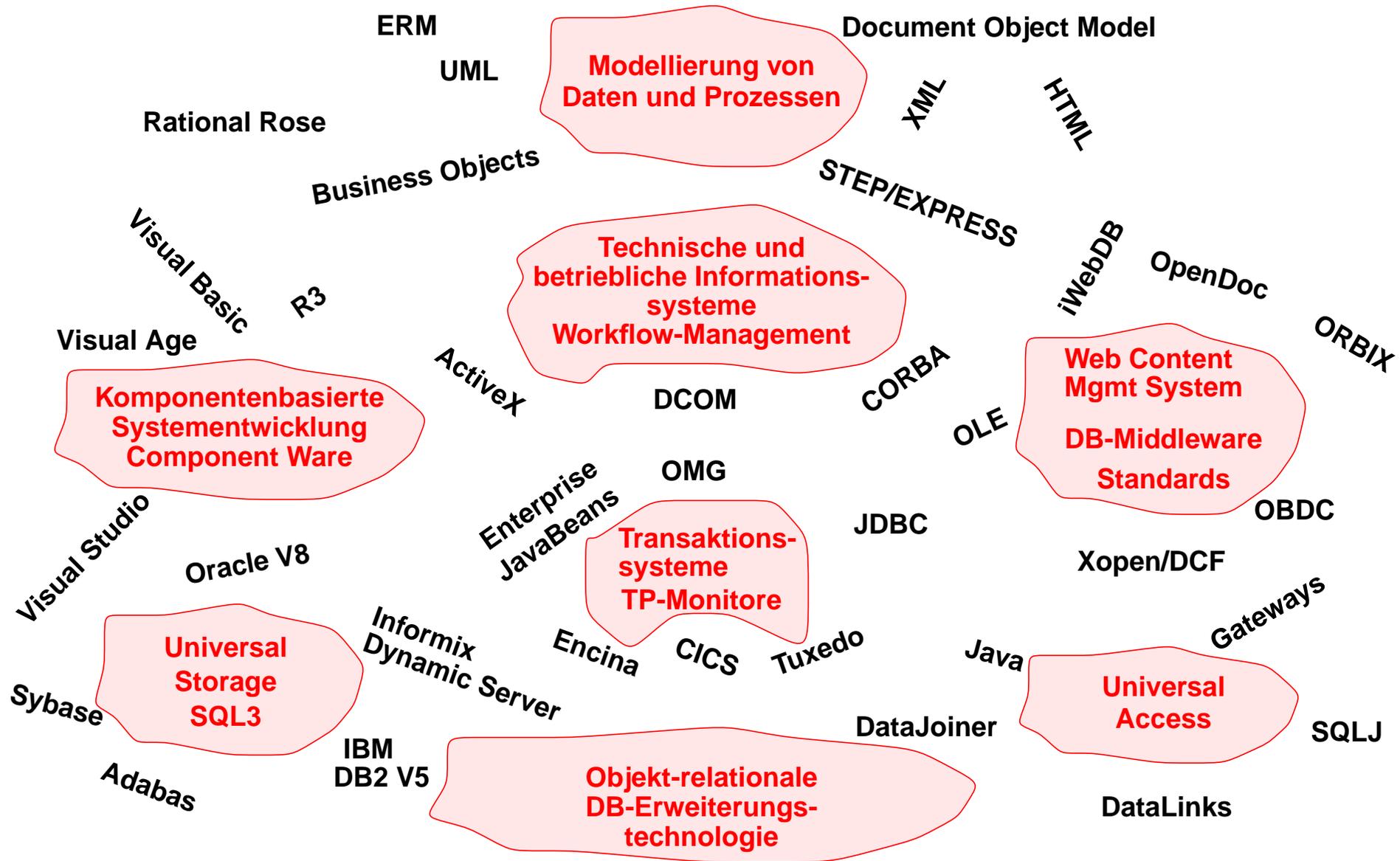
Ort: 48 - 210

Zeit: Mi., 11.45 - 13.15 Uhr

Übung:

Ort: noch festzulegen

Zeit: Do., 15.30 - 17.00 Uhr



Ziele

- **Vermittlung von Grundlagen- und Methodenwissen¹ zur Anwendung von Datenbanksystemen; Erwerb von Fähigkeiten und Fertigkeiten für DB-Administrator und DB-Anwendungsentwickler**
 - Entwurf, Aufbau und Wartung von Datenbanken, insbesondere auf der Basis von
 - Relationenmodell und SQL
 - objektorientierten und objekt-relationalen Datenmodellen mit Bezug auf die Standards ODMG und SQL:1999
 - Sicherung der Abläufe von DB-Programmen
 - Transaktionsverwaltung, Synchronisation, Fehlerbehandlung
 - Semantische Integrität, aktive DB-Mechanismen
 - Datenschutz und Zugriffskontrolle
 - Beschreibung und Analyse von wichtigen DB-Anwendungen
- **Voraussetzungen für Übernahme von Tätigkeiten:**
 - Entwicklung von datenbankgestützten Anwendungen
 - Nutzung von Datenbanken unter Verwendung von (interaktiven) Datenbanksprachen
 - Systemverantwortlicher für Datenbanksysteme, insbesondere Unternehmens-, Datenbank-, Anwendungs- und Datensicherungsadministrator

1. Grundlagenwissen ist hochgradig allgemeingültig und nicht von bestimmten Methoden abhängig. Die Halbwertszeit ist sehr hoch. Methodenwissen muß ständig an die aktuelle Entwicklung angepaßt werden. In der Informatik haben sich die entscheidenden Methoden alle 8-10 Jahre erheblich geändert. Werkzeugwissen ist methodenabhängig. Werkzeuge haben in der Informatik oft nur eine Lebensdauer von 2-3 Jahren.

ÜBERSICHT (vorl.)

0. Übersicht und Motivation

- Datenstrukturen und Datenbanken
- Voraussetzungen für die Vorlesung

1. Anforderungen und Beschreibungsmodelle

- Anforderungen an DBS
- Aufbau von DBS
- Beschreibungsmodelle
(Fünf-Schichten-Modell, Drei-Ebenen-Beschreibungsarchitektur)

2. Logischer DB-Entwurf

- Konzeptioneller DB-Entwurf
- Normalformenlehre (1NF, 2NF, 3NF, 4NF)
- Synthese von Relationen

3. Anwendungsprogrammier-Schnittstelle

- Schemaevolution
- Sichtenkonzept
- Kopplung mit einer Wirtssprache
- Übersetzung von DB-Anweisungen
- Eingebettetes / Dynamisches SQL, PSM

4. Transaktionsverwaltung

- Transaktionskonzept
- Ablauf von Transaktionen
- Commit-Protokolle

ÜBERSICHT (2)

5. Synchronisation

- Anomalien beim Mehrbenutzerbetrieb
- Theorie der Serialisierbarkeit
- Synchronisationsverfahren
(Sperrprotokolle, Optimistische Verfahren, Prädikatssperren, Mehrversions- und Zeitstempelfverfahren)
- Konsistenzebenen

6. Logging und Recovery

- Fehlermodell und Recovery-Arten
- Logging-Strategien
- Recovery-Konzepte - Abhängigkeiten
- Sicherungspunkte
- Transaktions-, Crash- und Medien-Recovery

7. Integritätskontrolle und aktives Verhalten

- Semantische Integritätskontrolle
- Regelverarbeitung in DBS
- Trigger-Konzept von SQL
- Definition und Ausführung von ECA-Regeln

8. Datenschutz und Zugriffskontrolle

- Technische Probleme des Datenschutzes
- Konzepte der Zugriffskontrolle
- Zugriffskontrolle in SQL
- Sicherheitsprobleme in statistischen DBs

ÜBERSICHT (3)

9. Objektorientierung und Datenbanken

- Beschränkungen klassischer Datenmodelle
- Grundkonzepte der Objektorientierung
- Einführung in ein OR-Datenmodell (SQL:1999)
- ORDBS: Anforderungen, Architekturvorschläge
- Erhöhung der Anfragemächtigkeit, Rekursion

10. Große Objekte

- Anforderungen und Verarbeitung mit SQL
- Lokator-Konzept
- Speicherungsstrukturen, . . .

11. Erweiterbares Typsystem

- Umbenannte Typen
- Benutzerdefinierte Datentypen/Funktionen
- Typ- und Tabellenhierarchien, . . .
- Konstruierte Typen

12. Anwendungsklassen

- Transaktionssysteme
- Data Warehouse
- Data Mining
- WWW-Anbindung von Datenbanken

LITERATURLISTE

- Chamberlin, D.:* DB2 Universal Database, Addison-Wesley Publ. Comp., 1999
- Date, C. J.:* An Introduction to Database Systems, Addison-Wesley Publ. Comp., Reading, Mass., 7th Edition, 2000
- Härder, T., Rahm, E.:* Datenbanksysteme – Konzepte und Techniken der Implementierung, Springer-Verlag, Berlin, 2001 (Es sind Hörerscheine erhältlich)
- Heuer, A., Saake, G.:* Datenbanken – Konzepte und Sprachen, 2. Auflage, Int. Thompson Publ. Comp., 2000
- Kemper, A., Eickler, A.:* Datenbanksysteme – Eine Einführung, 4. Auflage, Oldenbourg-Verlag, 2001
- Melton, J., Simon, A. R.:* SQL:1999 - Understanding Relational Language Components, Morgan Kaufmann Publishers, San Francisco, CA, 2002
- Ramakrishnan, R.:* Database Management Systems, McGraw-Hill, Boston, 1998.
- Weikum, G., Vossen, G.:* Transactional Information Systems, Morgan Kaufmann Publishers, San Francisco, CA, 2002

ZEITSCHRIFTEN:

- TODS* Transactions on Database Systems, ACM Publikation (vierteljährlich)
- Information Systems* Pergamon Press (6-mal jährlich)
- The VLDB Journal* (vierteljährlich)
- Informatik - Forschung und Entwicklung* (vierteljährlich)

TAGUNGSBÄNDE:

- SIGMOD* Tagungsband, jährliche Konferenz der ACM Special Interest Group on Management of Data
- VLDB* Tagungsband, jährliche Konferenz „Very Large Data Bases“
- IEEE* Tagungsband, jährliche Konferenz „Int. Conf. on Data Engineering“
- GI* Tagungsbände der Tagungen der Gesellschaft für Informatik, Tagungen innerhalb des Fachbereichs “Datenbanken und Informationssysteme”
- und viele weitere Konferenzreihen

Datenstrukturen ?

- **Bisher bekannte Datenstrukturen**

- Felder (Reihungen, Arrays,)
- Verbunde (Tupel, Sätze, Records,)
- Listen
- Graphen
- Bäume

→ bisher im **Hauptspeicher**,
d. h. Bestand nur für die Dauer einer Programmausführung
(„transiente“ Daten)

- **hier nun neue Aspekte:**

Nutzung von Hintergrundspeicher und neuen Operationen

- **Persistenz:**

Werte bleiben über Programmende, Sitzungsende,
Betriebssystem-Uptime, Rechnereinschaltung,
hinaus erhalten

- andere Arten des Zugriffs: Lese- und Schreiboperationen,
in vorgegebenen Einheiten von Blöcken und Sätzen

→ Strukturen und zugehörige Algorithmen (Suchen, Sortieren),
die im Hauptspeicher optimal sind, sind es auf Sekundärspeicher
nicht unbedingt!

- gezielter, wertabhängiger Zugriff auch auf sehr große Datenmengen
- umfangreiche Attributwerte (z. B. Bilder, Texte)

Datenmodelle ?

- **Mengen von Konstruktoren**

zur (abstrakten) Erzeugung von Datenstrukturen mit darauf definierten Operatoren

z. B. Tabellen (Relationen): Mengen von gleichartig strukturierten Tupeln

```
CREATE TABLE STUDENT
  ( MATRIKELNUMMER      INTEGER ,
    NACHNAME            VARCHAR ( 40 ) ,
    FBNUMMER            INTEGER ,
    GEBURTSDATUM        CHAR ( 8 ) ,
    . . . . . )
```

```
CREATE TABLE FACHBEREICH
  ( FBNUMMER            INTEGER ,
    FBNAME              VARCHAR ( 20 ) ,
    DEKAN               PROF ,
    . . . . . )
```

nicht nur ein einzelner Satz, sondern Menge

- **Wichtige Rolle von Beziehungen**

in einigen Datenmodellen auch sehr spezielle Arten von **Beziehungen** zwischen Sätzen und/oder Tupeln: Hierarchien, Zusammensetzungen, funktionale Zuordnungen u.v.a.m.

- **Modellbegriff**

vorgegebene Menge von (sprachlichen) Ausdrucksmitteln, mit denen Diskursbereich beschrieben (erfaßt) werden muß

→ auch Programmiersprachen und Betriebssysteme haben ein „Datenmodell“

Datenbanken ?

- **Große Datenmengen**

- auch, aber **nicht immer** entscheidend

- **Übereinstimmung von Modell und Miniwelt¹**

- Genauigkeit der Abbildung und Erhaltung ihrer Integrität (Bedeutungstreue)
- zeitgerechter und durch Integritätsbedingungen abgesicherter Änderungsdienst

- **Datenunabhängigkeit (der Anwendungen)**

- Benutzung der Daten, ohne Details der systemtechnischen Realisierung zu kennen (abstraktes „Datenmodell“, z. B. Tabellen)
- einfache Handhabung der Daten, mächtige Auswertungsoperationen

- **Offenheit der Daten für neue Anwendungen
(Anwendungsneutralität der Speicherung)**

- symmetrische Organisationsformen (keine Bevorzugung einer Verarbeitungs- und Auswertungsrichtung)
- explizite Darstellung der Annahmen/Zusicherungen (nicht in den Anwendungsprogrammen verstecken)
- Redundanzfreiheit aus der Sicht der Anwendung: keine wiederholte Speicherung in unterschiedlicher Form für verschiedene Anwendungen
- Konsistenzüberwachung durch das Datenbanksystem

1. Ein Datenbanksystem verwaltet Daten einer realen oder gedanklichen Anwendungswelt. Diese Daten gehen aus Informationen hervor, die stets aus den Sachverhalten und Vorgängen dieser Anwendungswelt durch gedankliche Abstraktionen (Abbilder, Modelle) gewonnen werden. Sie beziehen sich nur auf solche Aspekte des betrachteten Weltausschnitts, die für den Zweck der Anwendung relevant sind. Ein solcher Weltausschnitt wird auch als *Miniwelt* (Diskurswelt) bezeichnet.

Datenbanken ? (2)

- **Transaktionskonzept mit Garantie von ACID-Eigenschaften**

- Atomizität (*atomicity*)
- Konsistenz (*consistency*)
- Isolation (*isolated execution*)
- Dauerhaftigkeit (*durability*)

- **Ausfallsicherheit**

- Aufzeichnung redundanter Daten im Normalbetrieb
- Replikation von Datenstrukturen
- Vorkehrungen für den Katastrophenfall
- automatische Reparatur der Datenbestände nach Programm-, System- und Gerätefehlern
 - Rückgängigmachen unvollständiger Transaktionen, so daß sie wiederholt werden können
 - Wiederherstellen der Ergebnisse vollständiger Transaktionen, so daß sie nicht wiederholt werden müssen

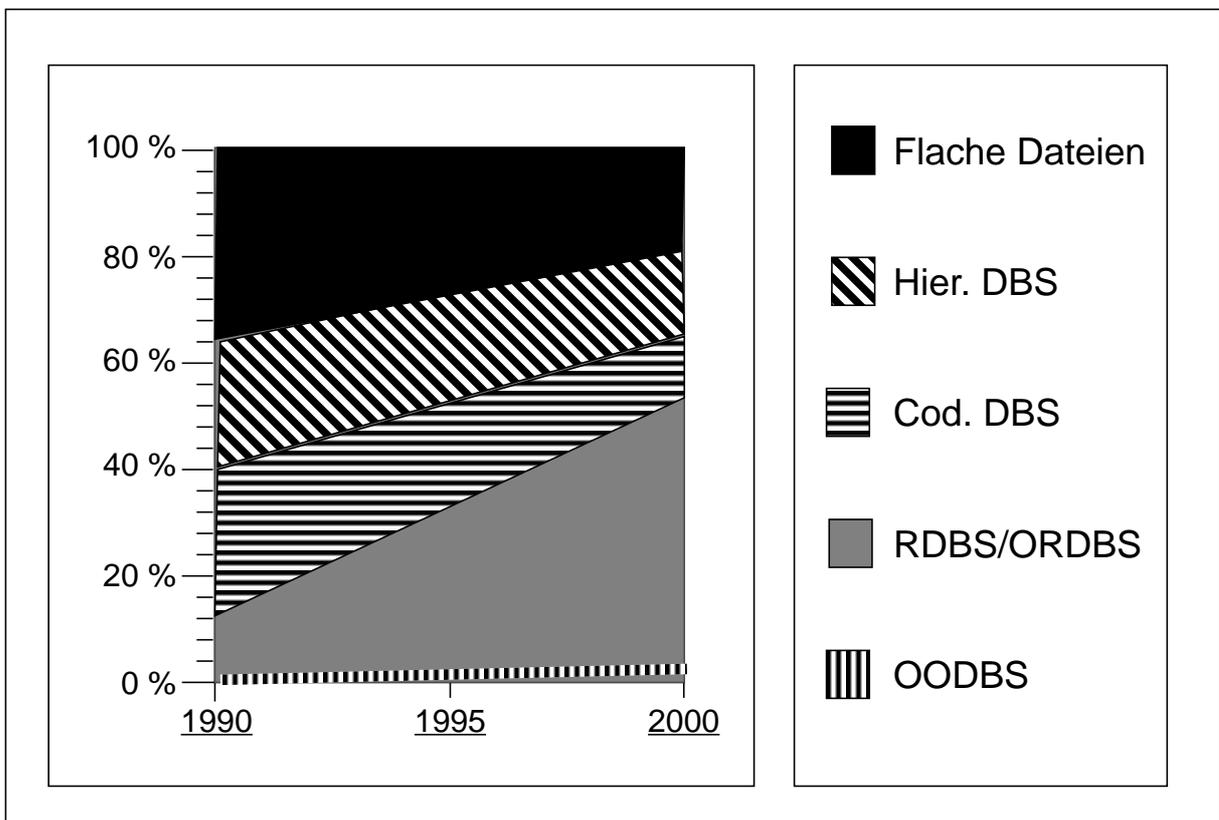
- **Mehrbenutzerbetrieb**

- gleichzeitiger (zeitlich eng verzahnter) Zugriff verschiedener Anwendungen und Benutzer auf gemeinsame Daten
- Synchronisation, d. h. Vermeidung von Fehlern in der wechselseitigen Beeinflussung
- Kooperation über gemeinsame Daten mit hohem Aktualitätsgrad

Verteilung von DBS und Dateien

- Es gibt verschiedenartige Datenmodelle und die sie realisierenden DBS
 - relational und objekt-relational (RDBS/ORDBS)
 - hierarchisch (DBS nach dem Hierarchiemodell)
 - netzwerkartig (DBS nach dem Codasyl-Standard)
 - objektorientiert (OODBS)

	<u>1990</u>	<u>1995</u>	<u>2000</u>
OODBs	1 %	2 %	3 %
RDBS/ORDBS	9 %	25 %	50 %
Cod. DBS	30 %	22 %	12 %
Hier. DBS	25 %	20 %	15 %
Flache Dateien	<u>35 %</u>	<u>31 %</u>	<u>20 %</u>
Gesamt	100 %	100 %	100 %

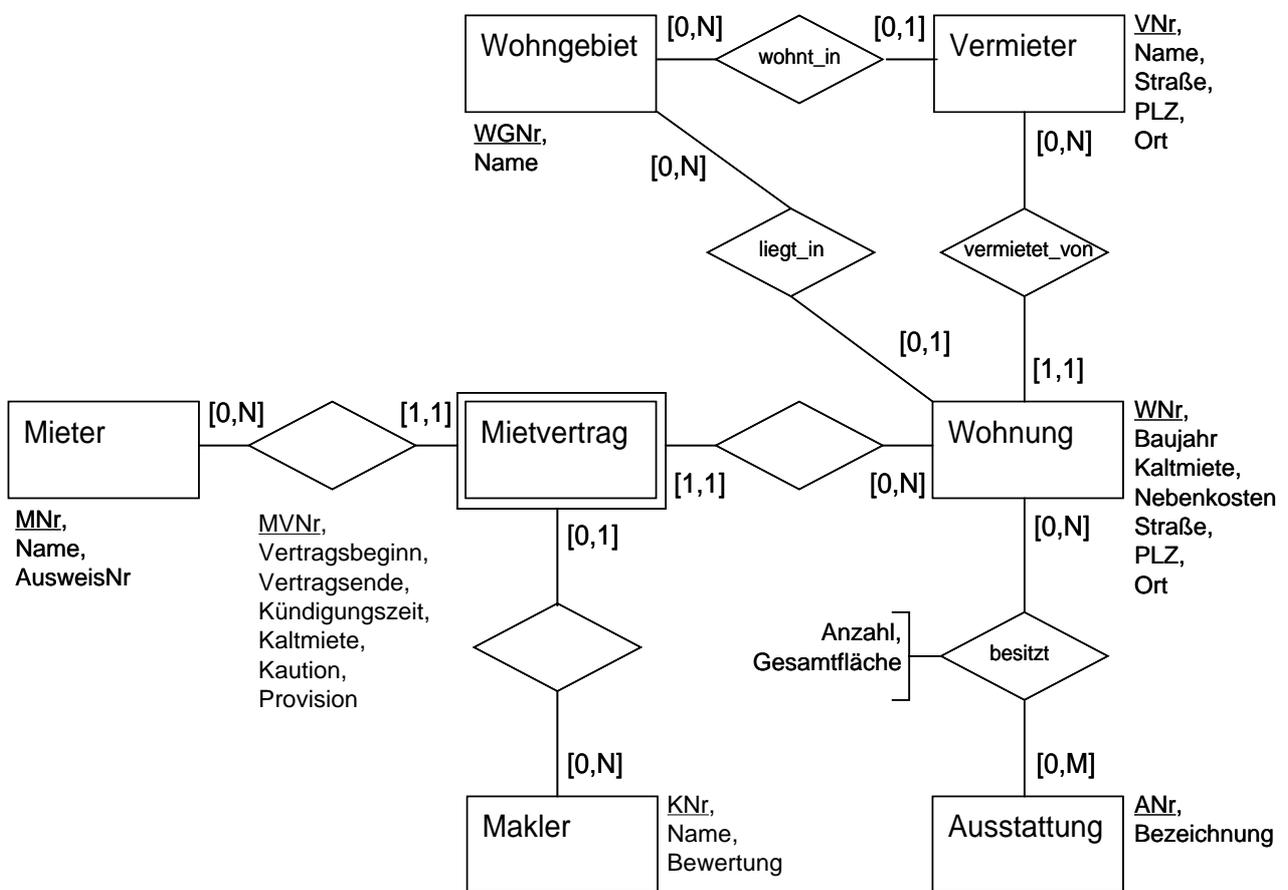


Voraussetzungen

- Beherrschung von

- Informationsmodellen (erweitertes ER-Modell)
- Relationenmodell und Relationenalgebra
- SQL-92 als Standardsprache

- Beispiel eines ER-Schemas



Voraussetzungen (2)

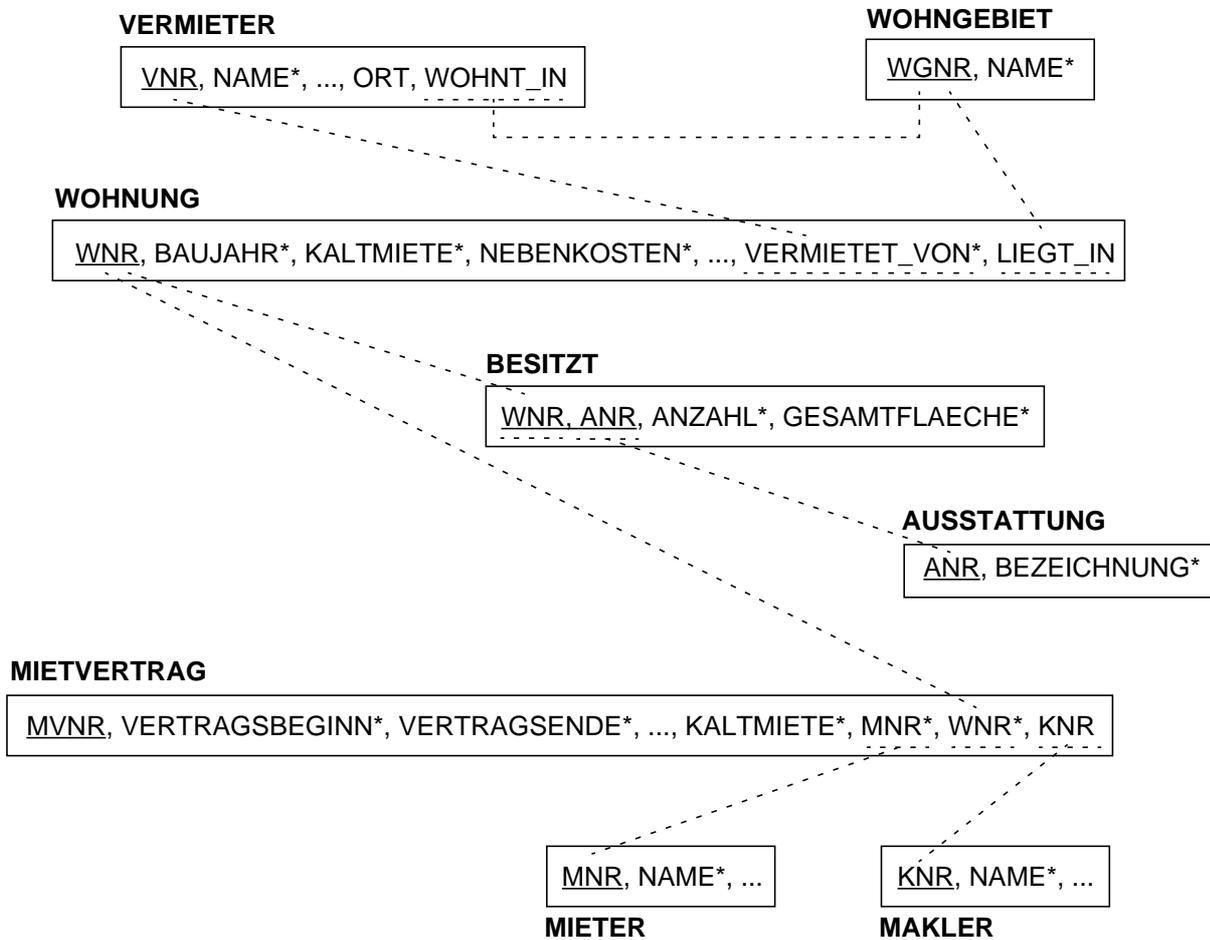
- Zugehöriges SQL-Schema

```
CREATE TABLE WOHNGBIET (  
    WGNR          WOHNGBIETSNUMMER  PRIMARY KEY,  
    NAME          CHAR(30)          NOT NULL);  
  
CREATE TABLE VERMIETER (  
    VNR          VERMIETERNUMMER    PRIMARY KEY,  
    NAME          CHAR(30)          NOT NULL,  
    STRASSE      CHAR(30),  
    PLZ          POSTLEITZAHL,  
    ORT          CHAR(30),  
    WOHT_IN     WOHNGBIETSNUMMER,  
    FOREIGN KEY (WOHT_IN) REFERENCES WOHNGBIET (WGNR));  
  
CREATE TABLE WOHNUNG (  
    WNR          WOHNUNGSNUMMER     PRIMARY KEY,  
    BAUJAHR      INT                DEFAULT 1800 NOT NULL  
                                CHECK (BAUJAHR >= 1800),  
    KALTMIETE    INT                NOT NULL,  
    NEBENKOSTEN  INT                NOT NULL,  
    STRASSE      CHAR(30),  
    PLZ          POSTLEITZAHL,  
    ORT          CHAR(30),  
    LIEGT_IN     WOHNGBIETSNUMMER,  
    VERMIETET_VON VERMIETERNUMMER  NOT NULL,  
    FOREIGN KEY (LIEGT_IN) REFERENCES WOHNGBIET (WGNR),  
    FOREIGN KEY (VERMIETET_VON) REFERENCES VERMIETER (VNR));  
  
CREATE TABLE AUSSTATTUNG (  
    ANR          AUSSTATTUNGSNUMMER PRIMARY KEY,  
    BEZEICHNUNG  CHAR(30)          NOT NULL);  
  
/* Zur Ausstattung gehören z. B. Schlafzimmer, Bad, Balkon, usw. */
```

Voraussetzungen (3)

```
CREATE TABLE BESITZT (  
    WNR            WOHNUNGSNUMMER,  
    ANR            AUSSTATTUNGSNUMMER,  
    ANZAHL         INT                DEFAULT 1 NOT NULL,  
    GESAMTFLAECHE INT                NOT NULL, /* In m2 */  
    PRIMARY KEY (WNR, ANR),  
    FOREIGN KEY (WNR) REFERENCES WOHNUNG (WNR),  
    FOREIGN KEY (ANR) REFERENCES AUSSTATTUNG (ANR));  
  
CREATE TABLE MAKLER (  
    KNR            MAKLERNUMMER        PRIMARY KEY,  
    NAME           CHAR(30)            NOT NULL,  
    BEWERTUNG      BEWERTUNGSSKALA);  
  
CREATE TABLE MIETER (  
    MNR            MIETERNUMMER        PRIMARY KEY,  
    NAME           CHAR(30)            NOT NULL,  
    AUSWEISNR     CHAR(30)            NOT NULL);  
  
CREATE TABLE MIETVERTRAG (  
    MVNR           MIETVERTRAGSNUMMER PRIMARY KEY,  
    VERTRAGSBEGINN DATE                NOT NULL,  
    VERTRAGSENDE  DATE                NOT NULL,  
    KUENDIGUNGSZEIT INT                DEFAULT 90, /* In Tagen */  
    KALTMIETE     INT                NOT NULL,  
    KAUTION       INT                DEFAULT 3 NOT NULL,  
    PROVISION     INT                DEFAULT 2 NOT NULL,  
    /* Werte fuer Kautiion und Provision seien n-Mal Kaltmiete */  
    MNR           MIETERNUMMER        NOT NULL,  
    WNR           WOHNUNGSNUMMER      NOT NULL,  
    KNR           MAKLERNUMMER        DEFAULT NULL,  
    FOREIGN KEY (MNR) REFERENCES MIETER (MNR) ON DELETE CASCADE,  
    FOREIGN KEY (WNR) REFERENCES WOHNUNG (WNR) ON DELETE CASCADE,  
    FOREIGN KEY (KNR) REFERENCES MAKLER (KNR)  
                                ON DELETE SET NULL);
```

Voraussetzungen (4)



* = NOT NULL

Voraussetzungen (5)

- Anfragen in SQL

- Lösche alle Vermieter, die keine einzige Wohnung vermieten und deren Wohngebiet unbekannt ist.

```
DELETE FROM VERMIETER V
WHERE NOT EXISTS
( SELECT *
  FROM WOHNUNG W
  WHERE W.VERMIETET_VON = V.VNR
)
AND V.WOHLT_IN IS NULL
```

- Erhöhe die Kaltmiete von Wohnungen, die später als 1990 gebaut wurden und im Uni-Wohngebiet in KL liegen, jeweils um 20% der zugehörigen Nebenkosten.

```
UPDATE WOHNUNG W
SET W.KALTMIETE = W.KALTMIETE + W.NEBENKOSTEN * 0.2
WHERE W.BAUJAHR > 1990
AND W.ORT = 'KL'
AND EXISTS
( SELECT *
  FROM WOHNGEBIET WG
  WHERE WG.WGNR = W.LIEGT_IN
  AND WG.NAME = 'Uni-Wohngebiet'
)
```

- Reduziere die Nebenkosten aller Wohnungen jeweils um 10%, die mindestens 2 Schlafzimmer oder eine Schlafzimmertgesamtfläche von $< 50 \text{ m}^2$ besitzen und deren Kaltmiete niedriger als ihre zugehörigen Nebenkosten ist.

```
UPDATE WOHNUNG W
SET W.NEBENKOSTEN = W.NEBENKOSTEN * 0.9
WHERE W.KALTMIETE < W.NEBENKOSTEN
AND EXISTS
( SELECT *
  FROM BESITZT B, AUSSTATTUNG A
  WHERE B.WNR = W.WNR
  AND B.ANR = A.ANR
  AND A.BEZEICHNUNG = 'Schlafzimmer'
  AND (B.ANZAHL >= 2 OR B.GESAMTFLAECHE < 50)
)
```

Voraussetzungen (6)

- Anfragen in SQL

- Finde alle zur Zeit vermieteten Wohnungen, deren Kaltmiete größer ist als die Gesamtsumme der vom (einzelnen) Mieter zu zahlenden Kaltmieten.
(Eine Wohnung kann mehrere gleichzeitig gültige Mietverträge haben (z. B. WG))

```
SELECT W.*
FROM WOHNUNG W
WHERE W.KALTMIETE >
      (SELECT SUM(MV.KALTMIETE)
       FROM MIETVERTRAG MV
       WHERE MV.WNR = W.WNR
       AND MV.VERTRAGSBEGINN <= TODAY
       AND MV.VERTRAGSENDE >= TODAY
      )
```

- Finde alle Wohnungen mit ihrer Gesamtfläche, die ein Bad und wenigstens ein Ausstattungsmerkmal mit der Gesamtfläche von 30 m² oder größer besitzen.

```
SELECT B.WNR, SUM(B.GESAMTFLAECHE)
FROM BESITZT B
WHERE EXISTS
      (SELECT *
       FROM AUSSTATTUNG A
       WHERE B.ANR = A.ANR
       AND A.BEZEICHNUNG = 'Bad'
      )
GROUP BY B.WNR
HAVING MAX(B.GESAMTFLAECHE) >= 30
```

- Finde alle Mieter, die zur Zeit mehr als eine Wohnung im Uni-Wohngebiet mieten.

```
SELECT M.*
FROM MIETER M
WHERE 1 <
      (SELECT COUNT(*)
       FROM MIETVERTRAG MV, WOHNUNG W, WOHNGBIET WG
       WHERE MV.MNR = M.MNR
       AND MV.WNR = W.WNR
       AND W.LIEGT_IN = WG.WGNR
       AND WG.NAME = 'Uni-Wohngebiet'
       AND MV.VERTRAGSBEGINN <= TODAY
       AND MV.VERTRAGSENDE >= TODAY
      )
```

Voraussetzungen (7)

- Anfragen in SQL

7. Finde alle Wohnungen, die 3 Schlafzimmer und 2 Bäder besitzen.

In SQL:

```
SELECT W.*
FROM WOHNUNG W
WHERE EXISTS
  (SELECT *
   FROM BESITZT B, AUSSTATTUNG A
   WHERE B.WNR = W.WNR
   AND B.ANR = A.ANR
   AND A.BEZEICHNUNG = 'Schlafzimmer'
   AND B.ANZAHL = 3
  )
AND EXISTS
  (SELECT *
   FROM BESITZT B, AUSSTATTUNG A
   WHERE B.WNR = W.WNR
   AND B.ANR = A.ANR
   AND A.BEZEICHNUNG = 'Bad'
   AND B.ANZAHL = 2
  )
```

Wie läßt sich diese Anfrage ohne EXISTS-Prädikate ausdrücken?

8. Finde alle Wohnungen, die alle Ausstattungsmerkmale haben.

In SQL:

```
SELECT W.*
FROM WOHNUNG W
WHERE NOT EXISTS
  (SELECT *
   FROM AUSSTATTUNG A
   WHERE NOT EXISTS
     (SELECT *
      FROM BESITZT B,
      WHERE B.WNR = W.WNR
      AND B.ANR = A.ANR
     )
  )
```

Alternative Formulierung: Finde alle Wohnungen, so daß es kein Ausstattungsmerkmal gibt, das die Wohnung nicht besitzt.

WWW-basiertes
Verarbeitungsmodell



Transaktions-
verarbeitung

Client/Server-
Verarbeitungsmodell



Objektorientiertes
Verarbeitungsmodell



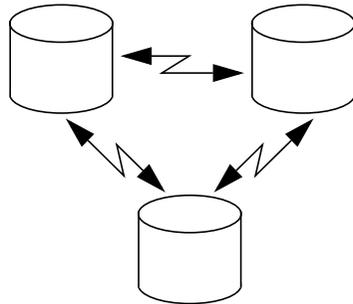
**Next-Generation
DBMS**

```

SELECT Unfall.Fahrer, Unfall.Vers-Nummer
FROM Unfall, Autobahn
WHERE CONTAINS(Unfall.Bericht, "Schaden"
                IN SAME SENTENCE AS
                ("schwer" AND "vordere" AND "Stoßstange"))
AND FARBE(Unfall.Foto, 'rot') > 0.6
AND ABSTAND(Unfall.Ort, Autobahn.Ausfahrt) < miles (0.5)
AND Autobahn.Nummer = A8;

```

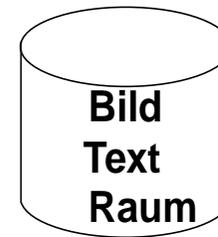
Parallele und Verteilte DBS



Aktive DBS



Multimedia-DBS



The Big Picture

A Dozen Long-Term Systems Research Problems (J. Gray)¹

1. Scalability:

Devise a software and hardware architecture that scales up by a factor for 10^6 . That is, an application's storage and processing capacity can automatically grow by a factor of a million, doing jobs faster (10^6 x speedup) or doing 10^6 larger jobs in the same time (10^6 x scaleup), just by adding more resources.

2. The Turing Test:

Build a computer system that wins the imitation game at least 30% of the time.

3. Speech to text:

Hear as well as a native speaker.

4. Text to speech:

Speak as well as a native speaker.

5. See as well as a person:

Recognize objects and motion.

6. Personal Memex:

Record everything a person and hears, and quickly retrieve any item on request.

7. World Memex:

Build a system that given a text corpus, can answer questions about the text and summarize the text as precisely and quickly as a human expert in that field. Do the same for music, art and cinema.

1. J. Gray is the recipient of the 1998 A. M. Turing Award. These problems, many related to database systems, are extracted from the text of the talk J. Gray gave in receipt of that award.

A Dozen Long-Term Systems Research Problems (J. Gray) (2)

8. **TelePresence:**

Simulate being some other place retrospectively as an observer (TeleObserver): hear and see as well as actually being there, and as well as a participant, and simulate being some other place as a participant (TelePresent): interacting with others and with the environment as though you are actually there.

9. **Trouble-Free Systems:**

Build a system used by millions of people each day and yet administered and managed by a single part-time person.

10. **Secure System:**

Assure that the system of problem 9 only services authorized users, service cannot be denied by unauthorized users, and information cannot be stolen (and prove it).

11. **AlwaysUp:**

Assure that the system is unavailable for less than one second per hundred years -- 8 9's of availability (and prove it).

12. **Automatic Programmer:**

Devise a specification language or user interface that:

- (a) makes it easy for people to express designs (1,000x easier),
- (b) computers can compile, and
- (c) can describe all applications (is complete).

The system should reason about application, asking questions about exception cases and incomplete specification. But it should not be onerous to use.

Menschliches Gedächtnis

- **Wieviel Information kann das menschliche Gehirn aufnehmen?**

- 200 MBytes an Information

T.K. Landauer: „How much do people remember? Some estimates of the quantity of learned information in long-term memory“, Cognitive Science 10 (4) pp. 477-493 (Oct-Dec 1986).

- Studie zur Aufnahme- und Vergessensrate sowie zur Nutzung bei normalen Tätigkeiten
- Bei 10^{12} - 10^{14} Neuronen kann man davon ausgehen, daß das Gehirn 10^3 - 10^5 Neuronen für jedes Bit des Gedächtnisses enthält
- Der größere Teil des Gehirns wird für die Wahrnehmung, Bewegungskontrolle usw. genutzt

→ Even if only 1 % of the brain is devoted to memory, it looks like your head accepts considerable storage inefficiency in order to be able to make effective use of the information (Landauer).

- **Mit der Landauer'schen „Schätzung“ und den ~ 6 Mrd Menschen auf Erden erhält man als gesamtes Speichervermögen aller menschlichen Gedächtnisse 1,200 PBytes.**

→ Rechner können heute alles digital speichern, was alle Menschen wissen oder an was sie sich erinnern können

- **Diese Schätzungen sind ungenau. Jedoch wäre es auch bei einem anderen „Aufnahmemodell“ kein Problem, die Informationsmenge eines Menschen auf eine Laptop-Platte zu speichern.**

- Menschliche Aufnahme- und Vergessensrate beim Langzeitgedächtnis: 1 Byte/sec nach Landauer
- Gesamtmenge in einem menschlichen Leben: 2 GBytes (2 Mrd Sek.)

Schlußfolgerungen

- **Es wird genug Platten- und Bandspeicher geben, um alles zu speichern, was alle Menschen schreiben, sagen, tun oder photographieren.**
 - Für das Schreiben gilt dies bereits heute
 - In einigen Jahren trifft das auch für die restlichen Informationen zu
 - Wie lange wird es noch dauern, alle VITA-Dokumente eines Menschen als Lebensgeschichte aufgezeichnet werden können?
- **Rechner speichern und verwalten Informationen besser und effektiver als Menschen**
 - Nach dem Jahr 2000 werden viele Platten und Kommunikationsverbindungen direkt Informationen aus Rechner-zu-Rechner- und nicht mehr (nur) aus Mensch-zu-Mensch-Kommunikation speichern
 - Wie lange wird es noch dauern, bis der Mensch die meiste gespeicherte Information gar nicht mehr zu sehen bekommt?
 - Wir müssen lernen, wie alles automatisch ausgewertet werden kann und was bei unserer knappen Zeit unserer besonderen Aufmerksamkeit bedarf.
- **Künftige Entwicklung**
 - Heute konzentriert man sich bei den „Digitalen Bibliotheken“ auf die Eingabe: auf das Scanning, Komprimieren und OCR von Informationen.
 - Morgen wird anstelle der Eingabe die „relevante Auswahl“ die wesentliche Rolle spielen: Selektion, Suche und Qualitätsbewertung von Informationen

→ **Wir können eine reale „World Encyclopedia“ mit einem echten „planetary memory for all mankind“ aufbauen, wie H.G. Wells bereits 1938 in seinem Buch „World Brain“ geschrieben hat!**