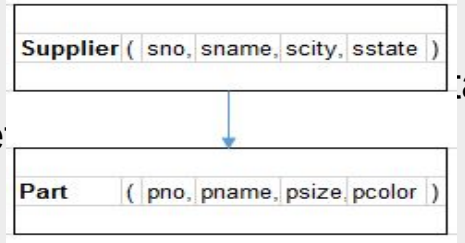
Jahre
olution der
tenmodelle

IMS - Hierarchischen Datenbankmodell

- Entitäten und deren Beziehungen werden mittels Graphentheorie dargestellt: Entitäten sind Knoten und Beziehungen Kanten.
- Die Datensätze sind in einer strengen Baumstruktur angeordnet.
- Es kann nur ein Schlüssel verwendet werden.
- Ein Datensatz ist genau über einen Vorgänger erreichbar.

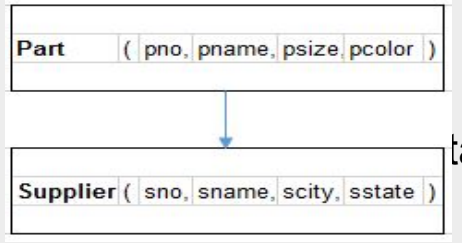
Detailbeispiel IMS

Schema 1:
Lieferanten zugeordnet



ät wird dem

Schema 2:
Ersatzteil zugeordnet.



tät wird dem

Query IMS

Im Schema 1 können somit alle roten Ersatzteile des Lieferanten #16 mittels folgender Abfrage aufgefunden werden:

```
Get unique Supplier (sno = 16)
Until no-more {
Get next within parent (color = red)
}
```

Die Abfrage nach dem roten Ersatzteil im Schema 1 könnte ebenfalls lauten:

```
Until no-more {
Get next Part (color = red)
}
```

Vorteile / Nachteile - IMS

Anwendern hat eine gewisse Vertrautheit mit hierarchischen Strukturen

Der Anwender muss die Struktur des Baumes kennen:
Jede Entität ist ausschließlich über die ihm zugewiesene Wurzel-Entität, die den Einstiegspunkt in die Hierarche darstellt, sowie über einen gerichteten Pfad erreichbar.

Änderungen am Datenbankdesign sind nur schwer durchführbar

Eine m:n-Beziehung zwischen Entitäten ist nur mit erheblichem Aufwand zu realisieren

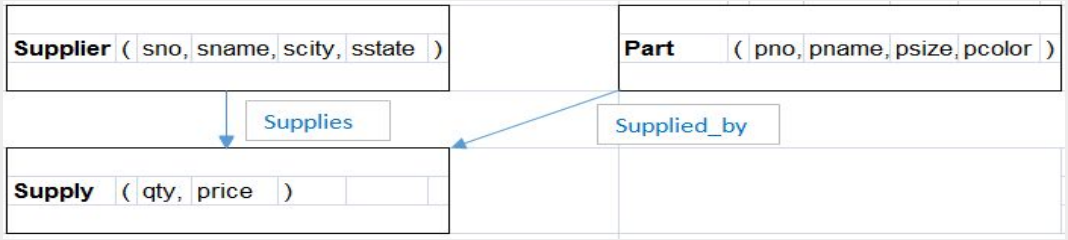
Integritätsverletzungen bei Operationen Löschen und Einfügen möglich

Im Falle einer Speicherüberschreitung sind aufwendige Reorganisationen der Datenbank nötig.

CODASYL - Netzwerkorientiertes Datenbankmodell

- Die Ablage von Daten erfolgt in Satzarten (record types).
- Satzarten können in beliebigen Beziehungen zueinander stehen (set type, owner-member).
- Beliebige Beziehungen zwischen Satztypen sind modellierbar.
- Grundlage zur Abbildung von komplexen Netzwerkstrukturen

Detailbeispiel CODASYL



Das netzwerkorientierte Datenbankmodell organisiert eine Sammlung von Datensatztypen, die jeweils einen eindeutigen Schlüssel aufweisen, mit Hilfe eines gerichteten Graphen.

Query CODASYL

Im Detailbeispiel CODASYL können alle roten Ersatzteile des Lieferanten #16 mittels folgender Abfrage aufgefunden werden:

```
Find Supplier (sno = 16)
Until no-more {
    Find next Supply record in Supplies
    Find owner Part record in Supplied_by
    Get current record
    check for red
}
```

Vorteile / Nachteile - CODASYL

Bietet eine redundanzfreie Abbildung von Datenstrukturen.

Unterschiedliche Möglichkeiten der Strukturierung stehen zur Verfügung.

Für die Modellierung von m:n Beziehungen wurden eindeutige Regeln definiert.

Ein performanter Zugriff wird mit Hilfe von eingerichteten Pfaden realisiert.

Die Darstellung beliebig komplexer Strukturen ist unübersichtlich.

Der Anwendungsprogrammierer muss über Kenntnisse des Zugriffspfades verfügen und der Anwender über Wissen der internen Datenbankstruktur.

Die Beziehungen zwischen den einzelnen Entitäten sind bereits beim Entwurf der Datenbank festzulegen.

Die Datenbank erfordert eine aufwendige Reorganisation im Falle einer

Litaraturverzeichnis

1. Michael Stonebraker, Joseph M. Hellerstein.
   What Goes Around Comes Around.
   https://people.cs.umass.edu/~yanlei/courses/CS691LL-f06/papers/SH05.pdf

2. Fachhochschule für Wirtschaft Berlin. (WS 2007).
   Grundlagen zur Verwendung von Datenbankmanagementsystemen
   http://userpage.fu-berlin.de/~schmiete/vorlesung/ws2007/dbs_part3_v1.pdf

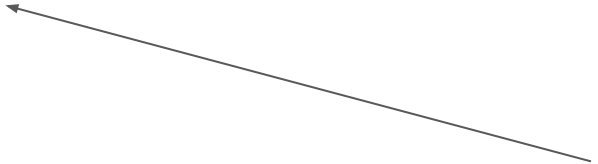
Relationale Datenbanken

Relation Abteilung

Name	Stiege	Stock	Raum	Telefon
Analyse	2	3	3-326	388-33-12
Modellierung	2	4	4-110	388-34-13

Relation Arbeiter

Nummer	Vor- Nachname	Geburtsdatum	Geschlecht	Dienststelle
12	Mark Zuckerberg	14.05.1984	M	Leiter
427	James Lebron	30.12.1984	M	Analytiker
732	Bill Gates	28.10.1955	M	Analytiker
1376	Hillary Clinton	26.10.1947	W	Modellierung



Name	Nummer
Analyse	732
Analyse	427
Analyse	12
Modellierung	1376

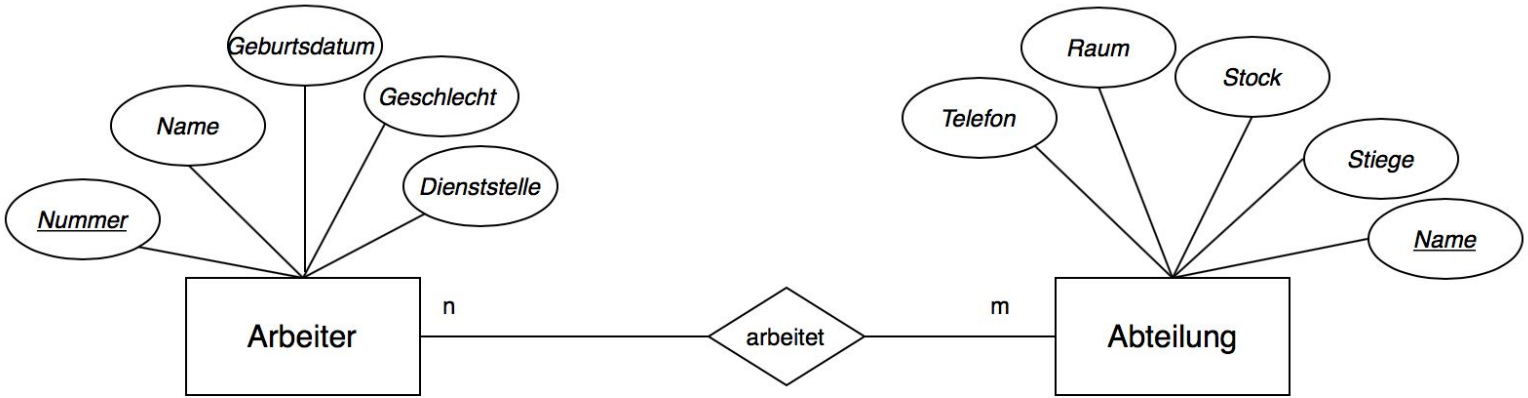
Relation Arbeiter in der Abteilung

Relational Era

- Das erste relationale Datenbanksystem war “System R” von IBM, mitte der 1970 Jahre.
- 1974 SEQUEL Sprache erfunden, die dann ins SQL umbenannt wurde.
- Es folgten Modifikationen: SQL-89, SQL-92 (SQL2), SQL-1999 (SQL3), SQL-2003, SQL-2006, SQL-2008.
- Ingres und Oracle auf dem Markt, 1979
- Sybase SQL Server im Jahr 1987
- Microsoft MySQL Server, 1989

Entity-Relationship Modell

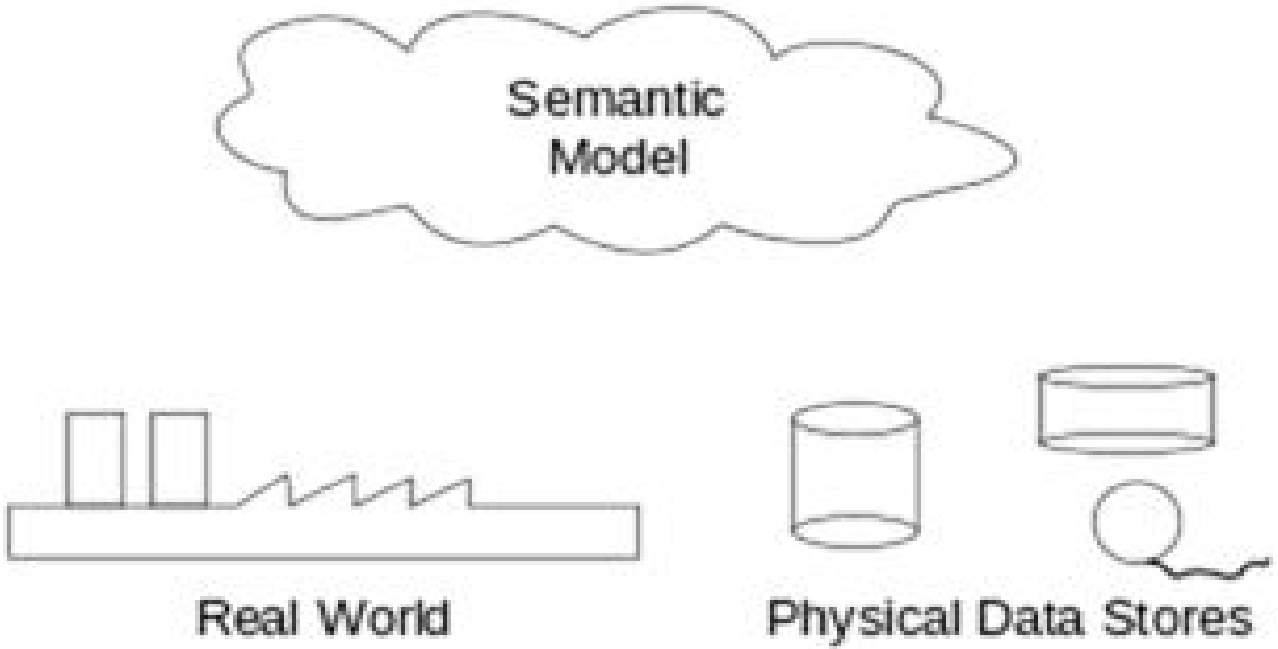
Beispiel mit Arbeiter und Abteilung:



Entity Relationship Era

- 1969: C.Bachmann hat seine ER-Notation entwickelt
- 1975: P. Chen hat eine alternative zum relationalen Datenmodell vorgeschlagen
- 1986: R. Barker hat die Ideen von Chen und Bachmann verwendet und seine eigene Notation entwickelt, die dann als Grundlage für weitere Entwicklungen diente
- ER Modell wurde nie als DBMS implementiert, da keine query language vorgeschlagen wurde.
- Bis heute in dem Datenbankdesign weit verbreitet

Semantik Data Model



ein konzeptionelles Datenmodell, das semantische Informationen enthält, die den Daten und den zwischen ihnen liegenden Beziehungen eine grundlegende Bedeutung verleihen

Klassifizierung - "instance_of" Beziehungen

Aggregation - "has_a" Beziehungen

Verallgemeinerung - "is_a" Beziehungen

Semi-Structured Era

strukturierte Daten

aber nicht in einem rationalen Modell, wie eine Tabelle oder ein Objekt-basierte Grafik organisiert

Arten von Semi-strukturierten Daten:

XML

JSON

Objektorientierung

Objektorientiertes Anwendungsprogramm

<=!=>

Relationales Datenbankmodell

Impedance Mismatch

- Struktur
- Instanz
- Kapselung
- Identität
- Arbeitsweise
- Organisatorisches

Lösung

- Objektorientierte Datenbanksystem
- Objektrelationales Mapping

Objektrelationale Datenbanksysteme

Relationale Datenbanken sind nicht für jeden Anwendungsfall ideal.

Beispiel: GIS

Objektrelationale Eigenschaften

- Große Objekt
- Mengenwertige Attribute
- Geschachtelte Relationen
- Typdeklarationen
- Referenzen
- Objektidentität
- Vererbung
- Benutzerdefinierte Operationen

Lösung

Postgres

SQL:1999

Vor- und Nachteile der Verwendung eines halbstrukturierten Datenformats

Vorteile

kann die Information einiger Datenquellen darstellen, die nicht durch Schema eingeschränkt werden können

bietet ein flexibles Format für den Datenaustausch zwischen verschiedenen Arten von Datenbanken

strukturierte Daten als semi-strukturierte (für Browsing-Zwecke) zu sehen

das Schema kann leicht geändert werden

Nachteile

keine Trennung zwischen den Daten und dem Schema gibt und die Menge der verwendeten Strukturen vom Zweck abhängt