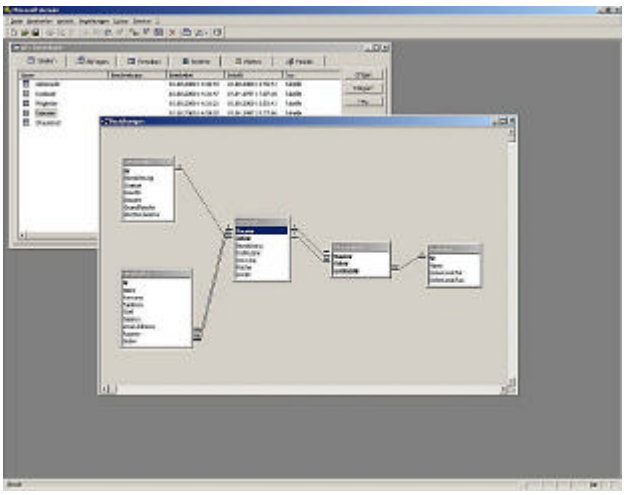
# Datenmodellierung[1]



## Inhaltsverzeichnis

1. Informationsstruktur ermitteln ... 2
2. Datenstruktur modellieren ... 3
2.1 Elemente des ER-Modells ... 3
2.1.1 Entities ... 3
2.1.2 Beziehungen zwischen Entities ... 4
2.1.3 Attribute ... 5
2.2 Umsetzung des semantischen Modells in ein logisches Modell ... 5
2.3 Identifikationsschlüssel ... 6
2.4 Entwicklung des relationalen Datenbankschemas mit Hilfe von Abbildungsregeln ... 6
2.5 Normalformen ... 7
2.5.1 Erste Normalform ... 7
2.5.2 Zweite Normalform ... 7
2.5.3 Dritte Normalform ... 8
3. Beispiel einer Datenbank ... 9
3.1 Einführung ... 9
3.2 Semantisches Modell ... 10
3.3 Zusammenfassung und relationales Modell ... 11

[1] Entnommen aus: Lehmann, Eberhard: Script *Einführungskurs Datenbank*, die Genehmigung des Autors zur Verwendung des Materials liegt vor

# 1. Informationsstruktur ermitteln

Für die Verwaltung der Klassen in einer Schule soll die Struktur einer Datenbasis entworfen und mit einem relationalen DBMS verwaltet werden. Das **Datenmodell**, das die für ein Informationssystem notwendigen Daten und Datenbeziehungen beschreibt wird in vier Phasen entwickelt.

**Externe Phase**
- Ermittlung des Informationsbedarfs der Nutzer
- Strukturierung dieser Informationen

**Konzeptionelle Phase**
Semantisches Modell: formale und strukturierte Beschreibung aller relevanten Objekte und deren Beziehungen untereinander.

**Logische Phase**
Umsetzung des semantischen Modells in ein relationales Datenbankmodell (logisches Modell).

**Physische Phase**
Modellierung der Datenbankstruktur mit einem relationalen DBMS (z.B. mit ACCESS).

Unabhängig vom DBMS

Abhängig vom DBMS

In der externen Phase wird der Informationsbedarf der Benutzer ermittelt und strukturiert. Es muss herausgefunden werden, welche Informationen das Datenbanksystem liefern soll (Output) und welche Informationen dafür bereitzustellen sind (Input). Aus dieser Analyse ergibt sich die Informationsstruktur. Der Input bildet die spätere Datenbasis der DB, der Output die zu erzielenden Ergebnisse, die Benutzersichten. Dabei lassen sich zwei grundsätzliche Ansätze unterscheiden:

**Top-down-Ansatz (globales Datenmodell)**

Die Informationsanforderungen aller späteren Datenbankbenutzer bestimmt die Informationsstruktur. Beispiel: Alle für die Schule relevanten Objekte (Schüler, Lehrer, Fächer, ...) werden erfasst. Es wird zunächst ein grobes Datenmodell entworfen und dann schrittweise verfeinert, so dass die einzelnen Anwendungen entstehen.

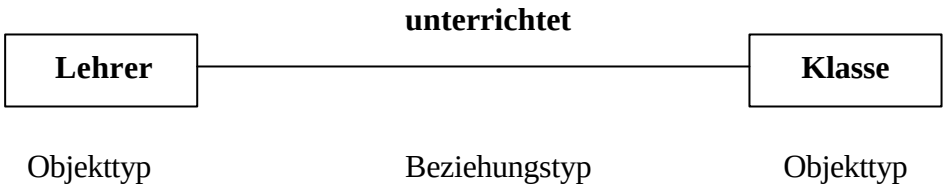
**Bottom-up-Ansatz**

Ein spezielles Problem ist Ausgangspunkt für die Datenbankentwicklung. Für die Lösung dieses Problems wird eine Anwendung entwickelt. Die Integration der einzelnen Anwendungen kann zu einem globalen Datenmodell führen.

Die Ermittlung der Informationsstruktur erfolgt anhand von Realitätsbeobachtungen. Betrachtet man z.B. die Miniwelt „Schule“ erkennt man gewisse **Objekte** wie das Fach *Französisch*, einen Schüler *Meier*, einen Lehrer *Müller*, einen Schulleiter *Direx* usw. Statt Objekt wird auch der Begriff **Entity** oder **Instanz** verwendet. Zwischen diesen Objekten bestehen Beziehungen, die bestimmte Abläufe (Prozesse) oder Abhängigkeiten in der Miniwelt darstellen.

Die Unterrichtsbelegung stellt die Beziehung zwischen einem Lehrer und den von ihm unterrichteten Klassen dar. Um zum Datenmodell zu gelangen, stellt man Beziehungen und Objekte mit gleichartigen Attributen zu Typen zusammen. Somit ergeben sich für das gewählte Beispiel

- die **Objekttypen** *Schüler, Lehrer* und *Klassen*
- der **Beziehungstyp** *unterrichtet*




Alle Lehrer bilden die **Objektmenge** *Lehrer*, alle Klassen die Objektmenge *Klasse* und alle Unterrichtsbelegungen die **Beziehungsmenge** *unterrichtet*.
Jeder Objekt- und Beziehungstyp definiert bestimmte **Eigenschaften (Merkmale, Attribute)**. Die Attribute des Objekttyps Lehrer sind: *Lehrer_ID, Nachname, Vorname, Fächerkombination.* Für den Objekttyp Klasse lassen sich folgende Eigenschaften feststellen: *Klassenbezeichung, Klassenleiter Lehrer_ID, Klassensprecher Schueler_ID.* Dem Beziehungstyp *unterrichtet* lassen sich z.B. die Merkmale *Lehrer_ID, Klassenbezeichnung, Fach* zuordnen.

## *2. Datenstruktur modellieren*

Um eine Datenbank aufzubauen, muss ein möglichst exaktes Abbild der relevanten Welt definiert werden, ein **Datenmodell**. Das Datenmodell soll den zu analysierenden Teilausschnitt der Welt (Miniwelt) mit ihrem vollständigen Bedeutungsgehalt (d.h. semantisch vollständig) abbilden. Dazu müssen alle relevanten Objekte und Beziehungen zwischen den Objekten in einem **semantischen Datenmodell** erfasst und beschrieben werden (**Konzeptionelle Phase**). Bei der Strukturierung, Bearbeitung und Auswertung der Daten wird mit dem **Entity-Relationship-Modell** (ER-Modell) gearbeitet. Anschließend wird eine Transformation in ein **relationales Modell** durchgeführt. Die Elemente des ER-Modells sind **Entities, Beziehungen zwischen Entities** und **Attribute** zur Charakterisierung der Entities.

### 2.1 Elemente des ER-Modells

### 2.1.1 Entities

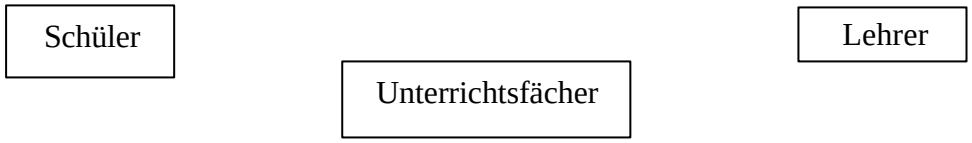
Ein Entity kann sein:

- ein Individuum (z.B. der Schüler *Meier*)
- ein reales Objekt (z.B. der Raum mit der Nummer *112*)
- ein abstraktes Konzept (z.B. der Kurs *Informatik*)
- ein Ereignis (z.B. eine *mündliche Prüfung*)

Definition: Ein **Entity** ist eine eindeutig identifizierbare Einheit.

Definition: Ein **Entity Set** fasst alle Entities zusammen, die durch gleiche Merkmale, nicht notwendigerweise aber durch gleiche Merkmalsausprägungen, charakterisiert sind.

Zur grafischen Darstellung von Entity Sets werden Rechtecke verwendet.




## 2.1.2 Beziehungen zwischen Entities

Eine Beziehung assoziiert wechselseitig zwei (oder mehrere) Entities. **Assoziation** bedeutet, dass ein Entity ein anderes Entity kennt und mit ihm in Wechselwirkung steht. Die **Kardinalität** einer Assoziation a(E1, E2) gibt an, wie viele Entities des Entity Sets E2 einem beliebigen Entity des Entity Sets E1 zugeordnet sein können.

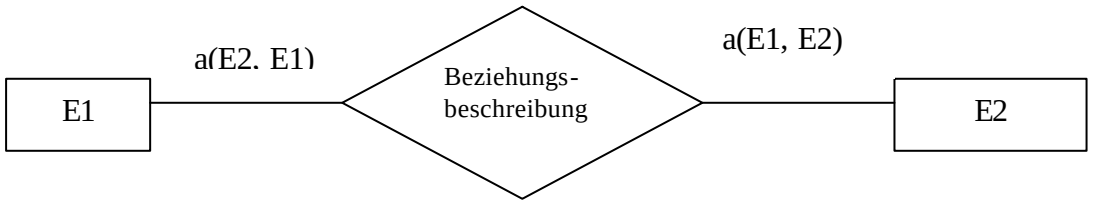
Definition: Eine **Beziehung** zwischen zwei Entity Sets E1 und E2 besteht aus der Assoziation a(E1, E2) und der dieser Assoziation entgegen gerichteten Assoziation a*(E2, E1).

Beispiel:

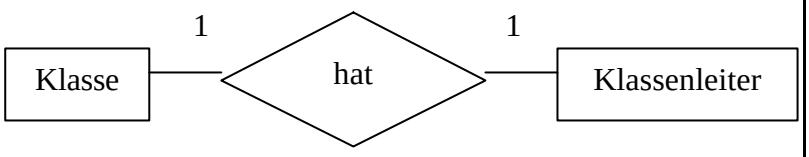
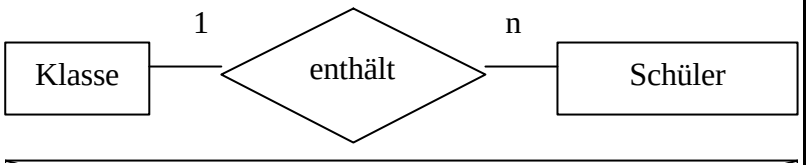
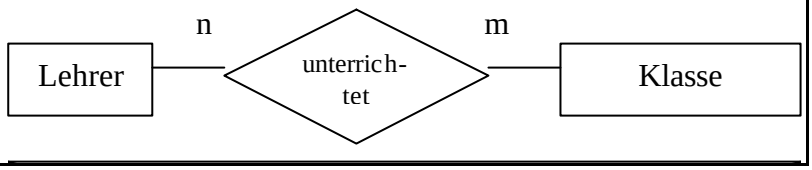
Eine Klasse kann von mehreren Schülern besucht werden: *a(Klasse, Schüler)*.
Ein Schüler besucht genau eine Klasse: *a*(Schüler, Klasse)*.

Definition: Die **Kardinalität** einer Assoziation a(E1, E2) gibt an, wie viele Entities des Entity Sets E2 einem beliebigen Entity des Entity Sets E1 zugeordnet sein können.

Zur grafischen Darstellung eines Beziehungstyps verwendet man im ER-Modell eine Raute. Dabei werden die Verbindungen zu den entsprechenden Entity Sets durch Linien dargestellt.
**Hinweis:** Für den Kardinalitätstyp *mehrfach* werden die Symbole m oder n verwendet.



Wichtig für die späteren Betrachtungen sind drei verschiedene Arten von Beziehungen:

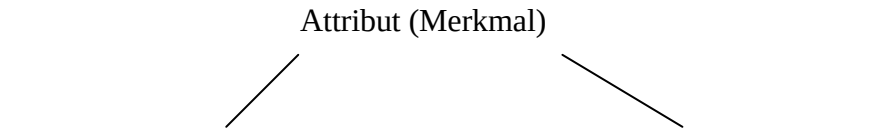
| Beziehungstyp | Beispiel |
|---|---|
| 1 – 1 |   |
| 1 – n |   |
| n – m |   |

## 2.1.3 Attribute

Interessierende Eigenschaften von Entities und Beziehungen lassen sich durch Attribute (Merkmale) erfassen. Auf der physikalischen Ebene wird statt Attribut auch der Begriff **Feld** (oder Datenfeld) verwendet.

Definition: Ein **Attribut** beschreibt eine bestimmte Eigenschaft, die sämtliche Entities eines Entity Sets oder sämtliche Einzelbeziehungen einer Beziehung aufweisen.

Wie bei einer Variablen ist bei einem Attribut zwischen seinem Namen und seinem Wert zu unterscheiden. Während der Name ein Attribut benennt und identifiziert, gibt der Wert die konkrete Ausprägung des Attributs für ein bestimmtes Entity oder Beziehung an. Welche diskreten Werte ein Attribut annehmen kann, legt sein Wertebereich fest.




Beispiel: *Name* *Meier*

## 2.2 Umsetzung des semantischen Modells in ein logisches Modell

Das semantische Modell hat zwei Konstruktionselemente: Entities und Beziehungen. Das logische Modell (relationales Modell) kennt nur ein Konstruktionselement: die Tabelle.

Der Übergang vom semantischen Modell zum logischen Modell ist nicht ohne weiteres möglich. Es kann vorkommen, dass das relationale Modell zwar 1 – 1 und 1 – n – Beziehungen unmittelbar abbilden kann, aber keine n – m – Beziehungen. Die im semantischen Modell definierten Klassen entsprechen deshalb in der Regel nicht der Struktur der Relationen (Tabellen) des logischen Datenbankschemas. Komplexe Beziehungen des Typs n – m können jedoch in 1 – n - Beziehungen aufgelöst werden, wenn für die Beziehungsmenge eine eigenständige Tabelle (Relation) definiert wird.
Eine feste Beziehung (Verknüpfung) zwischen den Relationen wird über Fremdschlüssel hergestellt. Der Fremdschlüssel muss in der anderen Relation ein Primärschlüssel (Identifikationsschlüssel) sein.

## 2.3 Identifikationsschlüssel

Jeder Datensatz einer Tabelle muss eindeutig identifizierbar sei. Dies kann durch ein Feld oder eine Kombination von Feldern gewährleistet werden.

Definition: Ein **Identifikationsschlüssel** besteht aus einem Attribut oder aus einer Kombination von Attributen, welcher jedes Entity eines Entity Sets eindeutig identifiziert.

Bei dem angegebenen Entity Set *Schüler* ist z.B. das Attribut *Schüler_ID* als Identifikationsschlüssel geeignet. Nicht geeignet ist dagegen ein anderes Attribut wie z.B. *Name*, da es für verschiedene Entities den gleichen Wert annehmen kann.
Gebräuchlich sind weiterhin noch die Begiffe Primärschlüssel und Sekundärschlüssel. Ein **Primärschlüssel** identifiziert eindeutig die Entities eines Entity Sets und ist daher zugleich auch ein Identifikationschlüssel. Dagegen ist ein **Fremdschlüssel** ein Attribut, dass für mehrere Entities den gleichen Wert annehmen kann. Ein Sekundärschlüssel dient also der Zusammenfassung von Entities zu Teilmengen mit einer gleichen Eigenschaft. Ein Fremdschlüssel FS in einer Tabelle 2 ist ein Feld oder eine Feldkombination, welcher in einer Tabelle 1 den Primärschlüssel PS bildet.

## 2.4 Entwicklung des relationalen Datenbankschemas mit Hilfe von Abbildungsregeln

Das semantische Modell kann im Einzelnen mit den nachfolgenden Abbildungsregeln in ein logisches Modell überführt werden.

| **Entity Set; Beziehung** | | **Abbildungsregel für das relationale Modell** |
|---|---|---|
| Entity Set | Regel 1 | Jedes Entity Set muss als eigenständige Relation definiert werden. |
| 1 : 1 – Beziehung | Regel 2 | Es genügt eine Relation |
| 1 : n – Beziehung | Regel 3 | Der Primärschlüssel der 1 – Relation ist Fremdschlüssel in der n – Relation. Es ist keine eigenständige Beziehungsmengen – Relation notwendig. |

| | | |
|---|---|---|
| n : m – Beziehung | Regel 4 | Jede komplex – komplexe Beziehungsmenge muss als eigenständige Relation definiert werden. Die Primärschlüssel der zugehörigen Entity Sets treten als Fremdschlüssel in der Beziehungsmengen – Relation auf. |

## 2.5 Normalformen

Relationen sollten so geplant werden, dass

- logische Widersprüche (Inkonsistenzen, Anomalien) in der Datenbasis und
- Redundanz (Mehrfachspeicherung gleicher Daten) vermieden sowie
- eine höchstmögliche Flexibilität und
- schneller Zugriff gewährleistet werden.

Deshalb sollten beim Entwurf von Relationen die nachfolgenden Normalformen NF beachtet werden.

### 2.5.1 Erste Normalform

Eine Relation ist in der **ersten Normalform**, wenn die Werte der Attribute elementar (atomar) sind.

Damit sind z.B. Mengen, Folgen und Arrays als Attributwerte ausgeschlossen. Elementar bedeutet, dass der Wert als Ganzes zu sehen ist und keine Teile des Wertes getrennt verarbeitet werden sollen.

Beispiel:

| Mitglieds_ID | Mitglied | Geburtsdatum |
|---|---|---|
| 1 | Ulrich Becker Tel.: 08234 / 123 XYZ<br>Maxweg 14<br>85408 Gammelsdorf | 01.02.73 |
| 2 | Julia Berger Tel.: 08123 / 456 ABC<br>Fischweg 22<br>85395 Altenkirchen | 02.04.80 |

Im obigen Beispiel sind die Werte des Attributs *Mitglied* nicht elementar. Das Attribut *Mitglied* sollte in die Attribute *Nachname, Vorname, Straße, PLZ, Ort* und *Telefon* aufgespaltet werden.

**Hinweis:** Die folgenden NF setzen die jeweils vorhergehende NF voraus!

### 2.5.2 Zweite Normalform

Die zweite NF ist nur dann relevant, wenn sich der Primärschlüssel aus mehreren Attributen zusammensetzt.

Eine Relation ist in der **zweiten Normalform**, wenn alle Nichtschlüssel – Attribute voll funktional abhängig vom Primärschlüssel sind.

Beispiel: Der Primärschlüssel wird durch die Attribute *Mitglieds_ID* und *Sport_ID* gebildet.

Relation *Abteilungszugehörigkeit* eines Sportvereins

| Mitglieds_ID | Sport_ID | Abteilungsname |
|---|---|---|
| 1 | 1 | Fußball |
| 1 | 2 | Handball |
| 2 | 3 | Schwimmen |
| 3 | 1 | Fußball |

Kritik:
Wäre der Abteilungsname vom zusammengesetzten Primärschlüssel *Mitglieds_ID, Sport_ID* voll funktional abhängig, dann müsste er einen anderen Wert annehmen, wenn sich der zusammengesetzte Primärschlüssel ändert. Dies ist jedoch nicht der Fall. Vielmehr ist der *Abteilungsname* nur von einem Teil des Primärschlüssels abhängig, nämlich der *Sport_ID.* Ändert sich die *Sport_ID*, dann ändert sich auch der *Abteilungsname.* Es besteht also keine vollständige funktionale Abhängigkeit des Attributs *Abteilungsname* vom Primärschlüssel. Die Relation *Abteilungszugehörigkeit* verstößt somit gegen die zweite Normalform.

Lösung:
Das Attribut *Abteilungsname* gehört zu einer eigenen Relation *Sportabteilungen* und nicht zur Relation *Abteilungszugehörigkeit* und muss aus ihr entfernt werden.

### 2.5.3 Dritte Normalform

Eine Relation befindet sich in der **dritten Normalform**, wenn alle Nichtschlüssel – Attribute voneinander funktional unabhängig sind.

Beispiel 1: Der Primärschlüssel ist die *Mitglieds_ID*

Relation *Mitglieder* eines Sportvereins

| Mitglieds_ID | Nachname | Geburtsdatum | Alter |
|---|---|---|---|
| 1 | Becker | 01.02.73 | 25 |
| 2 | Berger | 02.04.80 | 18 |
| 3 | Fischer | 05.06.81 | 17 |

Kritik:
Das Alter ist vom Geburtsdatum abhängig. Ändert sich der Zeitpunkt der Ansicht der Tabelle, so kann sich auch das Alter ändern (auf jeden Fall ein Jahr später). Das Geburtsdatum ist kein Primärschlüsselattribut und somit ein Nichtschlüsselattribut. Also ist das Alter von einem Nichtschlüssel – Attribut funktional abhängig. Damit liegt ein Verstoß gegen die dritte NF vor.
Lösung:
Das Alter kann aus dem Geburtsdatum jederzeit in einer Abfrage berechnet werden. Es ist also überflüssig, es als Attributwert zu speichern.

Beispiel 2:

| Mitglieds_ID | Nachname | Vorname | ... | Abteilung 1 | Abteilung 2 |
|---|---|---|---|---|---|
| 1 | Becker | Ulrich | ... | Fußball | Handball |
| 2 | Berger | Julia | ... | Schwimmen | Fußball |
| 3 | Fischer | Otto | ... | Leichtathletik | Fußball |

Kritik:
Die dritte NF ist zwar erfüllt, in der konkreten Anwendung kann diese Lösung jedoch zu Problemen führen:

- Ein Mitglied kann in mehr als zwei Abteilungen aktiv sein, so dass neue Spalten eingefügt werden müssen,
- Die Formulierung von Abfragen wird unnötig verkompliziert. Die Abfrage „Welche Mitglieder sind in der Fußball – Abteilung aktiv?“ wäre zu formulieren als „Bei welchen Mitgliedern ist Abteilung 1 = Fußball oder bei welchen Mitgliedern ist Abteilung 2 = Fußball?“

Lösung:
Für die Aufzählungswerte wird eine eigene Relation definiert:

Mitglieder

| Mitglieds_ID | Nachname |
|---|---|
| 1 | Becker |
| 2 | Berger |
| 3 | Fischer |

1 : n

Abteilungszugehörigkeit

| M_Nr | Sportart_Nr |
|---|---|
| 1 | 1 |
| 1 | 2 |
| 2 | 3 |
| 3 | 1 |
| 3 | 4 |

n : 1

Sportabteilungen

| Sport_ID | Abteilung |
|---|---|
| 1 | Fußball |
| 2 | Handball |
| 3 | Schwimmen |
| 4 | Leichtathletik |

# *3. Beispiel einer Datenbank*

## 3.1 Einführung

Die Verwaltung der Mitglieder eines Sportvereins soll mit Hilfe einer Datenbank gemanagt werden.
Tabellenstruktur mit Feldnamen:

| M_Nr | Vorname | Name | m/w | Straße | PLZ | Ort | Sportart | Beitrag |
|---|---|---|---|---|---|---|---|---|
| 1 | Ulrich | Becker | m | Maxweg 14 | 85408 | Gammelsdorf | S | 160,00 € |
| 2 | Ulrich | Becker | m | Maxweg 14 | 85408 | Gammelsdorf | H | 110,00 € |
| 3 | Julia | Berger | w | Fischweg 22 | 85395 | Altenkirchen | H | 110,00 € |
| 4 | Manuela | Friedmann | w | Bahnhofstr. 23 | 85406 | Zolling | L | 110,00 € |
| 5 | Otto | Fischer | m | Karlweg 12 | 85375 | Neufarn | F | 150,00 € |
| 6 | Georg | Frohmann | m | Meierweg 99 | 85408 | Daberg | F | 150,00 € |
| 7 | Hans | Huber | m | Postweg 12 | 85368 | Brugschlag | S | 160,00 € |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 8 | Hans | Huber | m | Postweg 12 | 85368 | Brugschlag | F | 150,00 € |

Dabei treten bei der Verwaltung der Vereinsmitglieder in einer Tabelle mehrere Probleme auf:

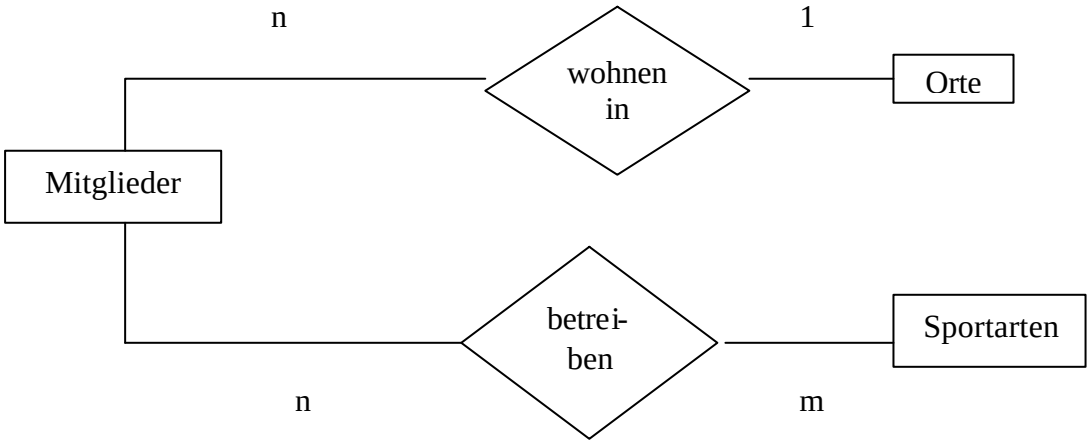
- Wenn ein Mitglied in mehreren Abteilungen aktiv ist, muss der gesamte Datensatz des Mitglieds erneut eingegeben werden (**Redundanz**). Dadurch ergibt sich bei der automatischen Erstellung der `M_Nr` das Problem, dass bei Mitgliedschaft in mehreren Abteilungen ein Mitglied verschiedene `M_Nr` erhält (**Inkonsistenz**). Außerdem ergibt sich hier eine Fehlerquelle bei erneuten Eingaben sowie bei Änderungen. Beim Ändern von Personendaten muss überprüft werden, in wie vielen Abteilungen das betreffende Mitglied aktiv ist, so dass die Änderungen vollständig durchgeführt werden können, da andernfalls unkorrekte Datensätze entstehen (**Änderungsanomalie**).
- Beim Löschen von Mitgliedern tritt das gleiche Problem auf wie oben (**Löschanomalie**).

Aus den eben genannten Gründen muss man sich eine andere Aufteilung der Daten überlegen, so dass die oben genannten Fehlerquellen bzw. Anomalien nicht auftreten können. Dazu erfasst man zunächst die Informationsstruktur des zu erstellenden Vereinsmodells:

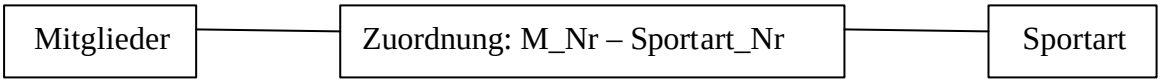
1. Folgende Daten sollen von den Mitgliedern verwaltet werden: Mitglieds-Nr, Vorname, Nachname, Geburtsdatum, Geschlecht, Straße, PLZ, Wohnort, betriebene Sportart sowie der zu zahlende Abteilungsbeitrag.
2. Ein Mitglied kann in mehreren Abteilungen aktiv sein. In einer Sportabteilung können aber auch mehrere Mitglieder aktiv sein.
3. Von jedem Mitglied wird nur eine Adresse (Hauptwohnsitz) erfasst. In einem Ort können mehrere Mitglieder wohnen.

## 3.2 Semantisches Modell

Auf Grund der im vorhergehenden Abschnitt beschriebenen Informationsstruktur ergibt sich das folgende semantische Datenmodell:




Die Beziehung *Mitglieder wohnen in Orte* ist eine 1 : n- Beziehung, also ist keine eigene Beziehungsmengen – Relation notwendig (Regel 3).
Die Beziehung *Mitglieder betreiben Sportarten* ist eine n : m – Beziehung, es ist also eine eigene Beziehungsmengen – Relation notwendig (Regel 4). Folglich wird für die Beziehungsmenge *betreiben* eine eigene Relation *Zuordnung: M_Nr – Sportart_Nr* konstruiert werden.

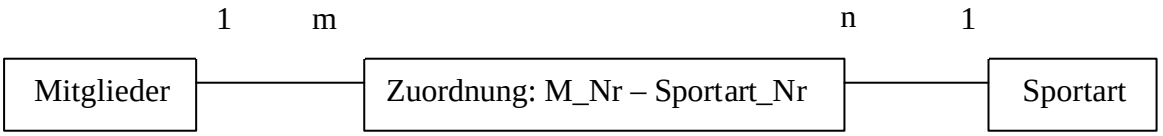



Im nächsten Schritt sind die sich aus der Informationsstruktur ergebenden Kardinalitätstypen der Assoziationen zu ermitteln:

**Beziehung *Mitglieder – Zuordnung: M_Nr – Sportart_Nr***
a(Mitglieder, Zuordnung M_Nr – Sportart_Nr) → Wie viele Sportarten kann ein bestimmtes Mitglied betreiben? → ein oder mehrere → Kardinalität: n
a*(Zuordnung M_Nr – Sportart_Nr, Mitglieder) → Wie oft kann ein bestimmtes Mitglied eine Sportart betreiben? → einmal → Kardinalität: 1
Fazit: Die Beziehung *Mitglieder – Zuordnung: M_Nr – Sportart_Nr* ist eine 1 : n – Beziehung; es ist keine eigene Beziehungsmengen – Relation notwendig (Regel 3).

**Beziehung *Zuordnung: M_Nr – Sportart_Nr – Sportarten***
a(Zuordnung M_Nr – Sportart_Nr, Sportarten) → Wie oft kann eine Sportart von einem bestimmten Mitglied betrieben werden? → einmal → Kardinalität: 1
a*(Sportarten, Zuordnung M_Nr – Sportart_Nr) → Von wie vielen Mitgliedern kann eine Sportart betrieben werden? → ein oder mehrere → Kardinalität: n
Fazit: Die Beziehung *Zuordnung: M_Nr – Sportart_Nr – Sportarten* ist eine 1 : n – Beziehung; es ist keine eigene Beziehungsmengen – Relation notwendig (Regel 3).

**Zusammenfassung:**
Insgesamt wurde die n : m- Beziehung *Mitglieder betreiben Sportarten* in zwei 1 : n - Beziehungen aufgelöst. Die Beziehungsmengen – Relation *Zuordnung: M_Nr – Sportart_Nr* muss auf der logischen Ebene als eigenständige Tabelle dargestellt werden:




## 3.3 Zusammenfassung und relationales Modell

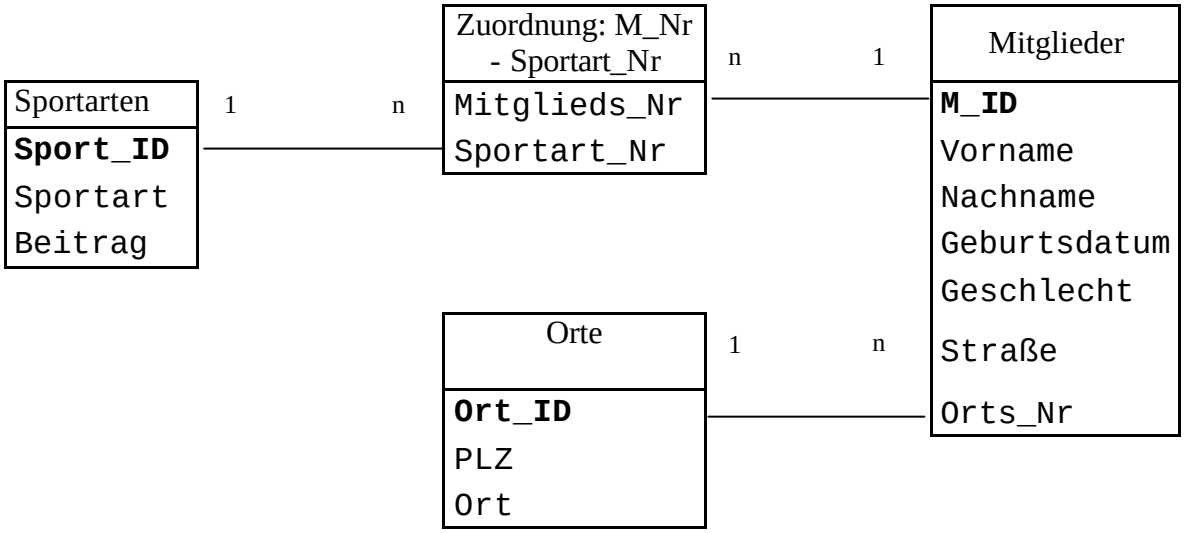
**Regel 1:** Jedes Entity Set muss als eigenständige Tabelle mit einem eindeutigen Primärschlüssel definiert werden.

| Mitglieder |
|---|
| **M_ID** |
| Nachname |
| Vorname |
| Geburtsdatum |
| Geschlecht |
| Strasse |

| Orte |
|---|
| **Ort_ID** |
| PLZ |
| Ort |

| Sportarten |
|---|
| **Sport_ID** |
| Sportart |
| Beitrag |

**Regel 4:** Bei m : n – Beziehungen muss eine eigenständige Beziehungsmengen – Relation definiert werden. Die Primärschlüssel der zugehörigen Entity Sets treten als Fremdschlüssel in der Beziehungsmengen – Relation auf.

| Zuordnung: M_Nr – Sportart_Nr |
|---|
| Mitglieds_Nr |
| Sportart_Nr |

**Regel 3:** Für alle 1 : 1 und 1 : n – Beziehungen gilt: Eine eigene Beziehungsmengen - Relation ist nicht notwendig. Der Primärschlüssel der 1 – Relation ist in der n – Relation (hier Orts_Nr) aufzunehmen.