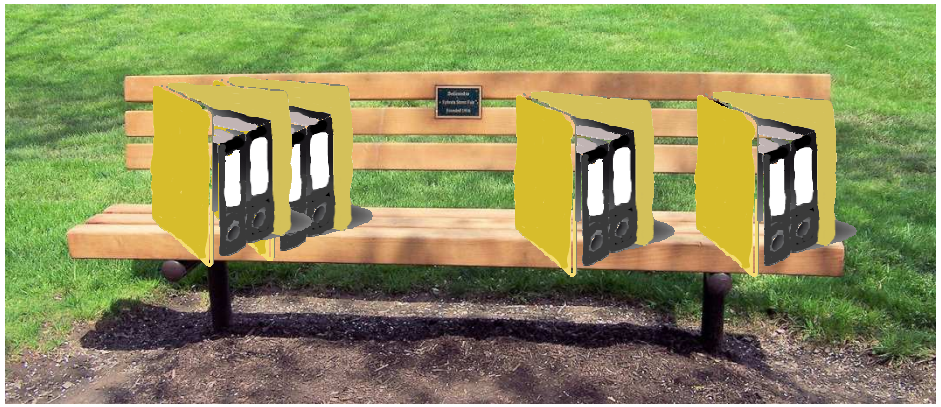
Daten – Bank



3. Vorlesung

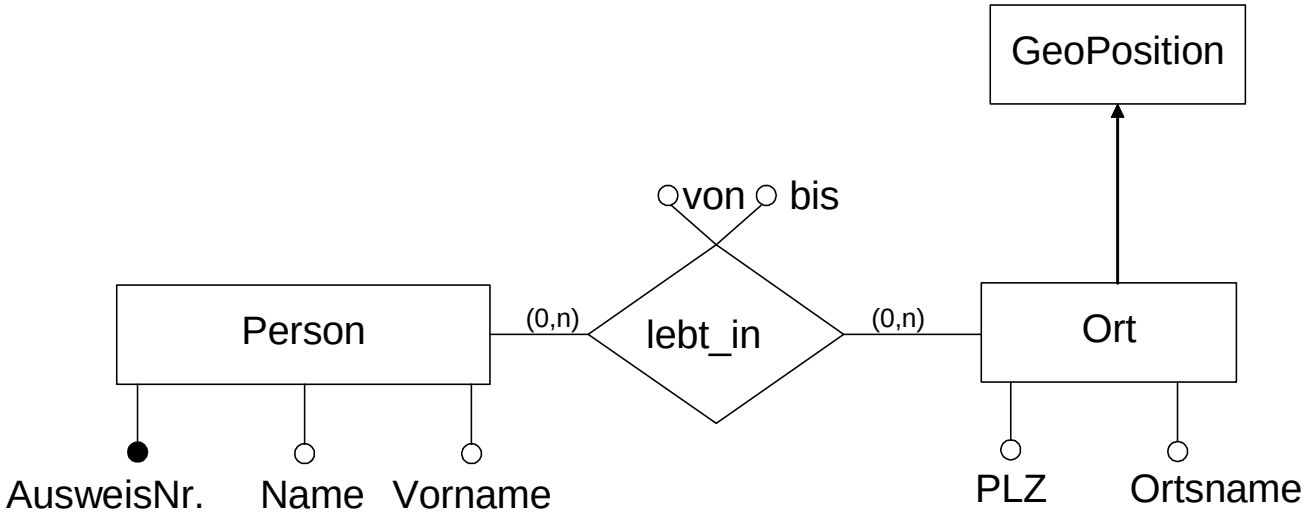
Repetitorium:

- Wer: Tung Le Trong
- Wann: 21.7.2017 (Freitag)
- Genauer wann …: von **10-18 Uhr**
- Wo: H IV

Bisher …

SQL:

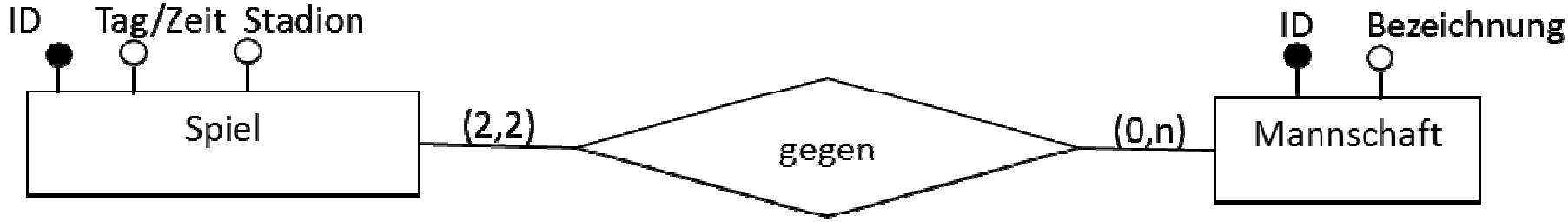
- create
- insert
- select



Übung 2 – ER-Diagr. für folgende Aussage erstellen

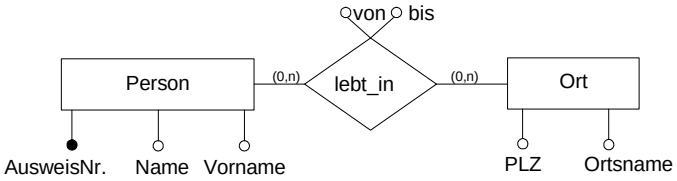
- Aussage:
  - Bei einem Spiel spielen zwei Mannschaften an einem bestimmten Tag und Uhrzeit in einem Stadion gegeneinander.

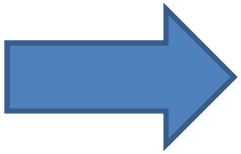
… eine weitere Möglichkeit!

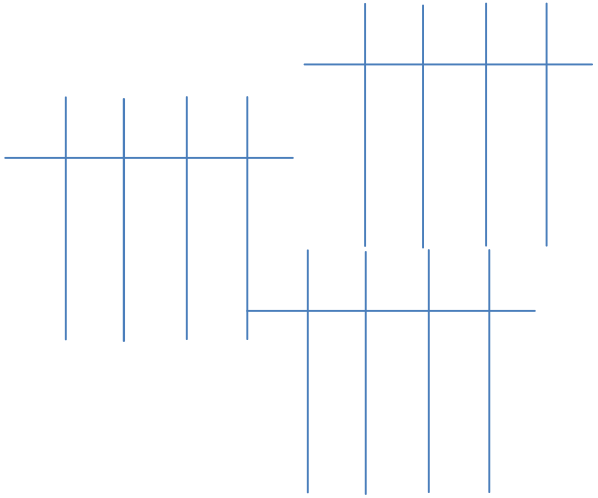


Bem.: So wird verhindert, dass eine Mannschaft gegen sich selbst spielt.
Wenn man bei Spiel einen zusammengesetzten Schlüssel wählt (Tag/Zeit/Stadion) kann man auch modellieren, dass nicht zwei Spiele gleichzeitig im gleichen Stadion stattfinden.

ER-Modell gegeben







Frage 1: Wie daraus Tabellen (Relationen) bauen?

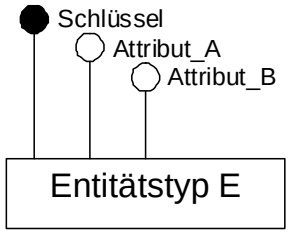
Frage 2: Welche Alternativen habe ich?

ER-Abbildung zu Relationen

Entitätstypen

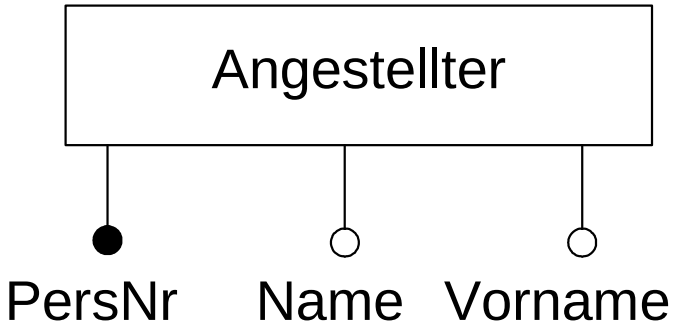
- Ein Entitätstyp wird zu einer Relation (Tabelle), dessen Relationenschema aus allen Attributen des Entitätstyps besteht.
- Jedes Tupel der Tabelle entspricht dann genau einer Entität des Entitätstyps.
- Etwaige Schlüssel werden übernommen und üblicherweise an den Anfang des Relationenschemas gestellt und unterstrichen.

„Regel“:



E (<u>Schlüssel</u>, Attribut_A, Attribut_B)

Beispiel

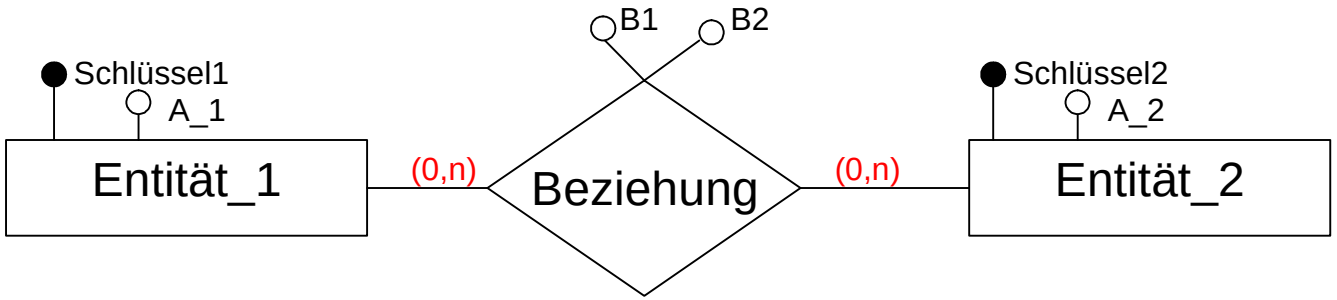


ANGESTELLTER (<u>PersNr</u>, Name, Vorname)

ANGESTELLTER

<u>PersNr</u>	Name	Vorname
001	Jon	Lucas
003	Jon	Smith
103	Lucas	Jon

Beziehungstyp



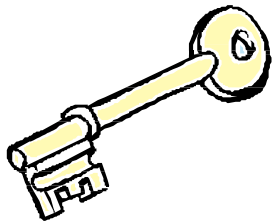
ENTITÄT_1 (<u>Schlüssel1</u>, A_1)

ENTITÄT_2 (<u>Schlüssel2</u>, A_2)

BEZIEHUNG (Schlüssel1, Schlüssel2, B1, B2)

Schlüssel für die Tabelle/Relation Beziehung?

Schlüssel



- Ein ***Schlüssel*** identifiziert eine Entität. Er besteht aus einer Menge von Attributen, deren Werte alle Instanzen einer Entität eindeutig bestimmen. (aus ER!)

- Ein ***Schlüssel*** (key) einer Relation ***r(R)*** ist eine minimale Teilmenge ***K*** von ***R***, so dass für je zwei **verschiedene** Tupel $t_1, t_2 \in r$ gilt:
  - $t_1(K) \neq t_2(K)$ und
  - keine **echte** Teilmenge K' von K hat diese Eigenschaft.

- Ein Schlüssel kann als Integritätsbedingung angesehen werden. Falls K Schlüssel von r(R), $t_1 \in r$, $t_1(K) = t_2(K)$, $t_1 \neq t_2$ dann darf t_2 nicht in r(R) eingefügt werden.

Schlüssel?

Raum

Name
NM103
NM117
NM123
SR9

Tutorium

TutorName	Raum	Wochentag	Startzeit
Sadik	NM117	Montag	08:15
Alina	NM117	Montag	12:15
Max	NM117	Donnerstag	14:15
Max	SR9	Donnerstag	12:15

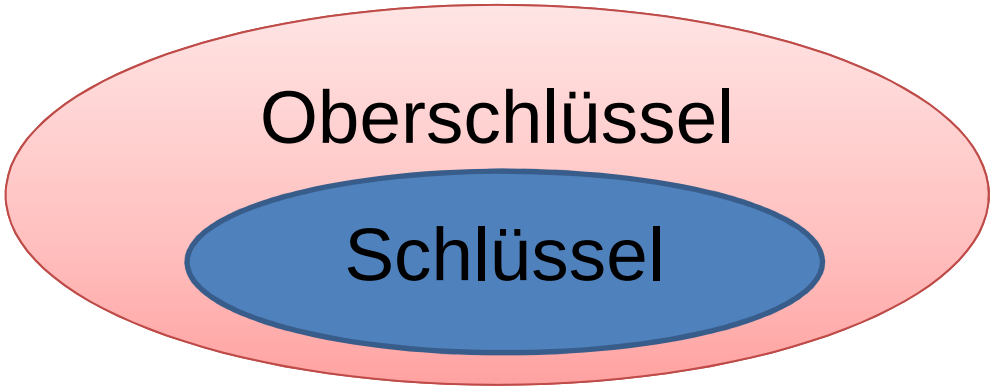
Ein Schlüssel kann als Integritätsbedingung angesehen werden.

Falls K Schlüssel von r(R), $t_1 \in r$, $t_1(K) = t_2(K)$, $t_1 \neq t_2$ dann dürfte t_2 nicht in r(R) eingefügt werden.

Oberschlüssel

- ***K*** ist ein ***Oberschlüssel*** (super key) der Relation, falls ***K*** einen Schlüssel enthält.

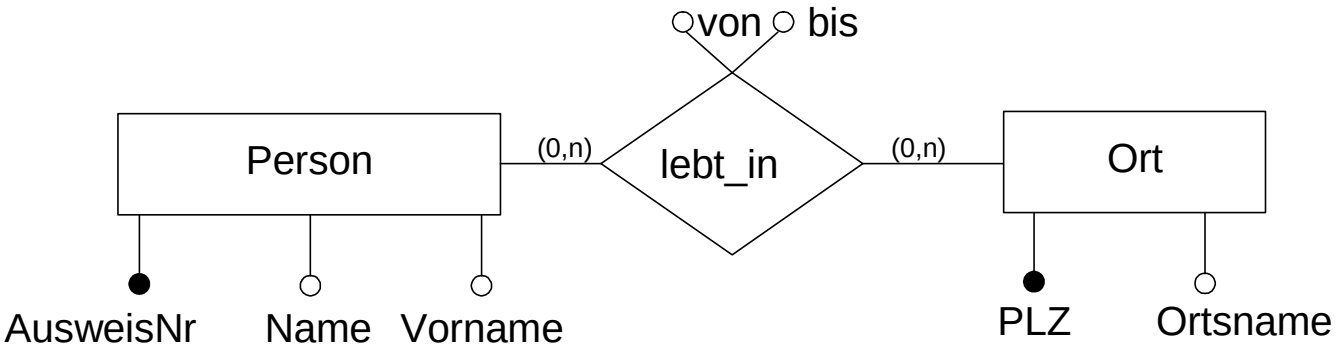
- … also aus Schlüssel → Oberschlüssel (aber nicht umgekehrt)



Wichtig für Normalformen (später)!

- **Eine Relation kann mehrere Schlüssel besitzen.** Man spricht dann auch von **Schlüsselkandidaten**.
- Im Allgemeinen wird **ein** Schlüssel als **Primärschlüssel** ausgezeichnet. Dieser wird im Relationenschema durch Unterstreichen gekennzeichnet.

Beispiel



PERSON	(<u>AusweisNr</u>, Name, Vorname)
ORT	(<u>PLZ</u>, Ortsname)
LEBT_IN	(AusweisNr, PLZ, von, bis)

Schlüssel für LEBT_IN?

Beispiel mit Instanzen

PERSON

AusweisNr	Name	Vorname
001	Jon	Lucas
003	Jon	Smith
103	Lucas	Jon

ORT

PLZ	Ortsname
501	Buli
503	Wali
603	Kali

LEBT_IN

AusweisNr	PLZ	von	bis
001	501	23.12.2000	25.12.2010
003	501	17.08.2004	Null
001	503	01.01.1999	01.01.2012

Jon Lucas (001) lebt(e)_in Buli (501), vom 23.12.2000 bis zum 25.12.2010!

PERSON

AusweisNr	Name	Vorname
001	Jon	Lucas
003	Jon	Smith
103	Lucas	Jon

ORT

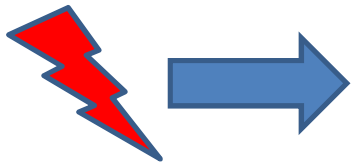
PLZ	Ortsname
501	Buli
503	Wali
603	Kali

LEBT_IN

AusweisNr	PLZ	von	bis
001	501	23.12.2000	25.12.2010
003	501	17.08.2004	Null
001	503	01.01.1999	01.01.2012
001	501	01.01.2012	Null

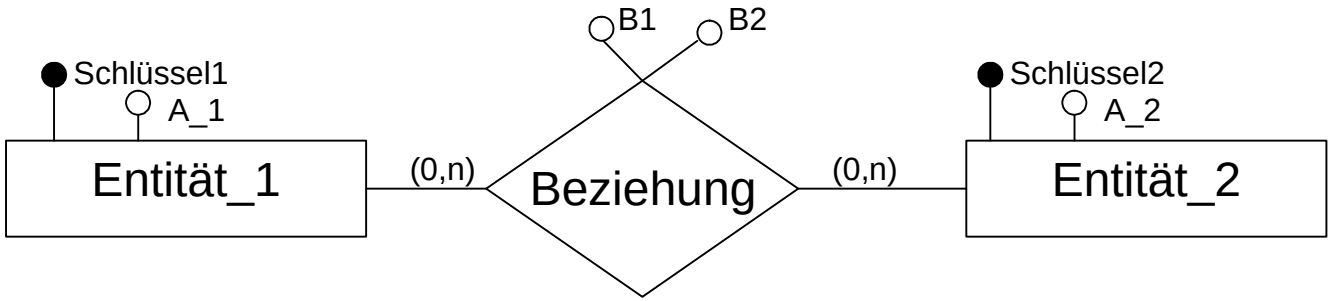
Jon Lucas (001) lebt(e)_in Buli (501), vom 23.12.2000 bis zum 25.12.2010!

Jon Lucas (001) lebt(e)_in Buli (501), vom 01.01.2012 bis heute!



Attribut „von“ der Relation muss auch Teil des Schlüssels sein!

(0,n) und (0,n)



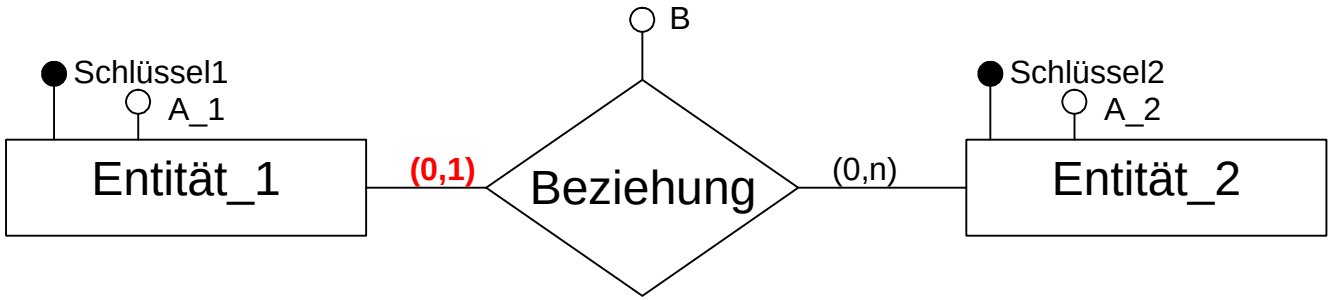
ENTITÄT_1 (<u>Schlüssel1</u>, A_1)

ENTITÄT_2 (<u>Schlüssel2</u>, A_2)

BEZIEHUNG (<u>Schlüssel1</u>, <u>Schlüssel2</u>, <u>B1</u>, B2)

Je nach Situation Teil des Schlüssels oder nicht!

(0,1) und (0,n)

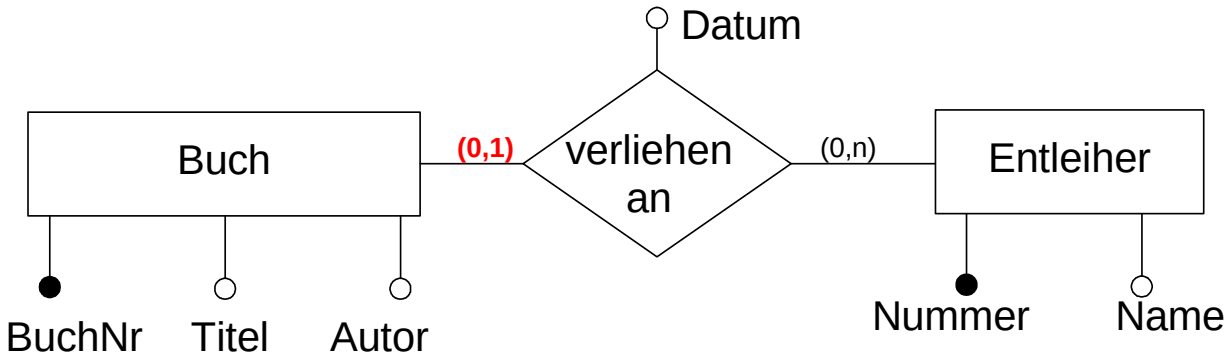


ENTITÄT_1 (Schlüssel1, A_1)

ENTITÄT_2 (Schlüssel2, A_2)

BEZIEHUNG (**Schlüssel1**, Schlüssel2, B)

Beispiel



⬇

BUCH (<u>BuchNr</u>, Titel, Autor)
ENTLEIHER (<u>Nummer</u>, Name)
VERLIEHEN_AN(<u>BuchNr</u>, Nummer, Datum)

Beispiel mit Instanzen

BUCH

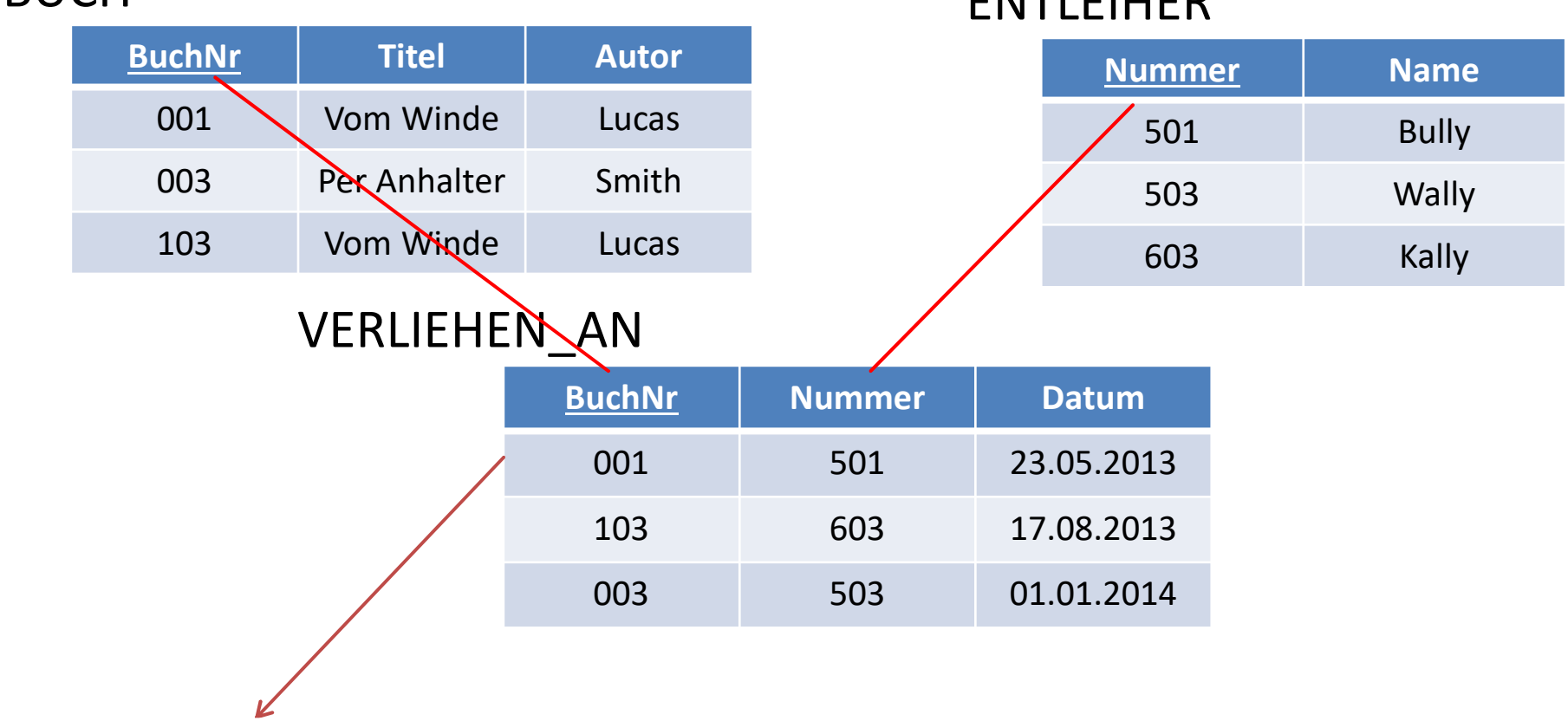
BuchNr	Titel	Autor
001	Vom Winde	Lucas
003	Per Anhalter	Smith
103	Vom Winde	Lucas

ENTLEIHER

Nummer	Name
501	Bully
503	Wally
603	Kally

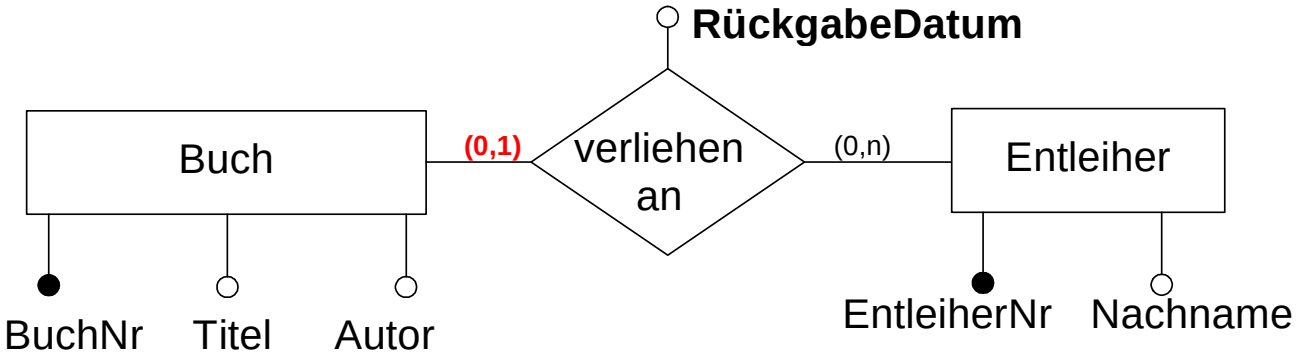
VERLIEHEN_AN

BuchNr	Nummer	Datum
001	501	23.05.2013
103	603	17.08.2013
003	503	01.01.2014



Das Buch (001) wurde verliehen an Bully, Datum 23.05.2013.

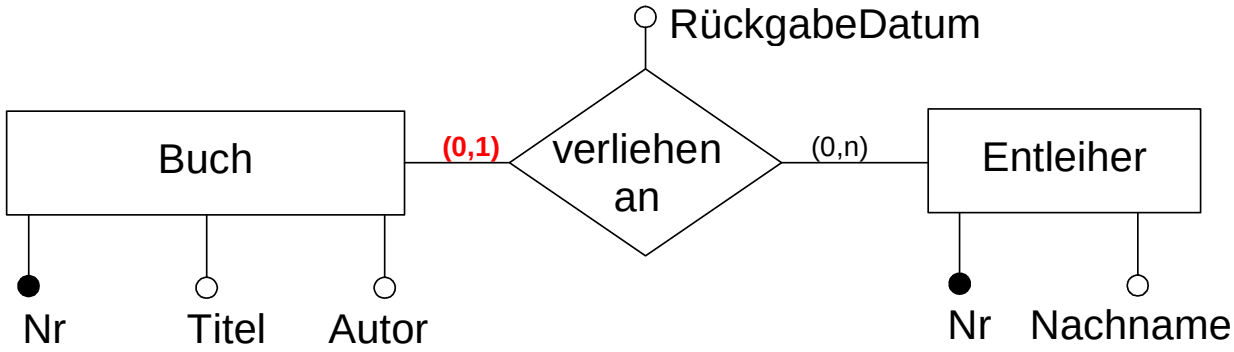
Einheitliche und ausdrucksstarke Schreibweisen beibehalten …



⬇

BUCH (<u>BuchNr</u>, Titel, Autor)
ENTLEIHER (<u>EntleiherNr</u>, Name)
VERLIEHEN_AN(<u>BuchNr</u>, EntleiherNr, RückgabeDatum)

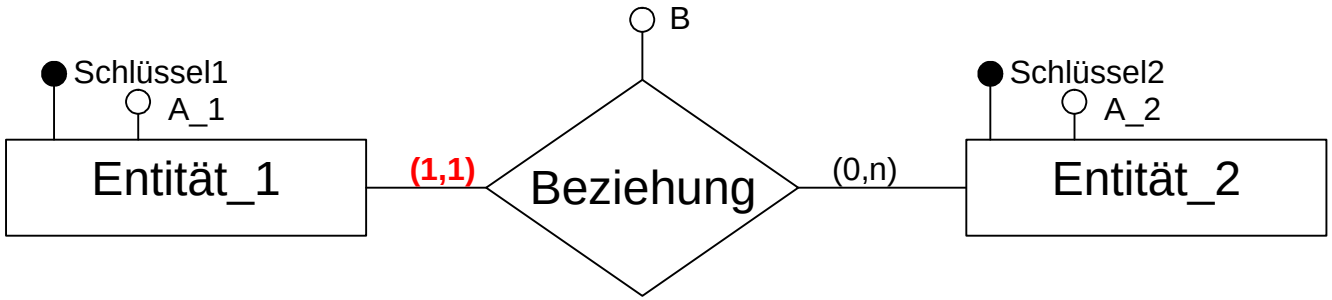
… oder auch so …



⬇

BUCH (Nr, Titel, Autor)
ENTLEIHER (Nr, Name)
VERLIEHEN_AN(BuchNr, EntleiherNr, RückgabeDatum)

(1,1)

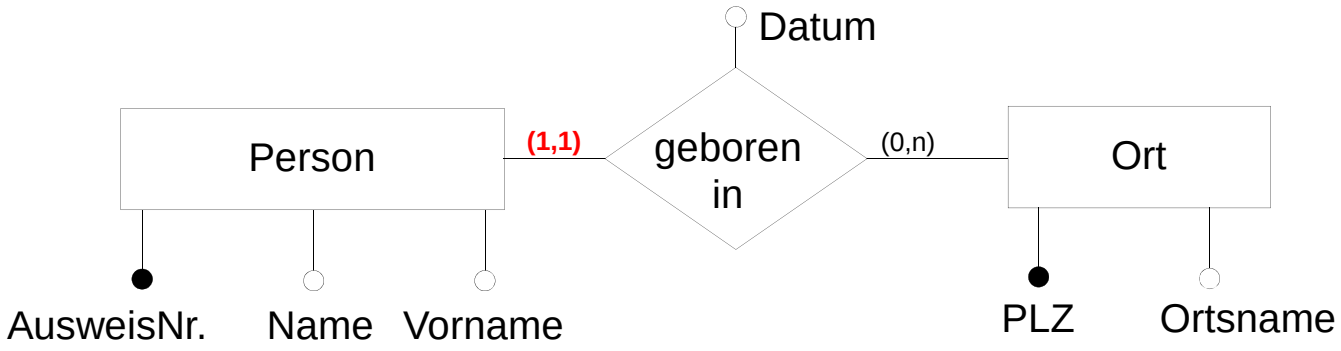


ENTITÄT_1 (Schlüssel1, A_1, Schlüssel2, B)

ENTITÄT_2 (Schlüssel2, A_2)

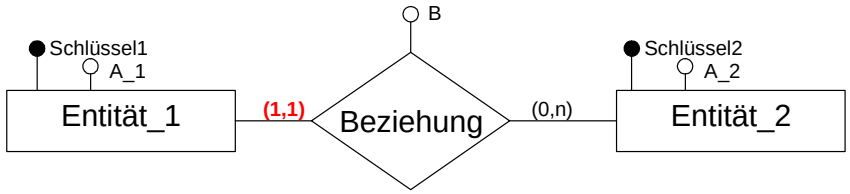
Hier sind nur noch zwei Relationen notwendig!

Beispiel



PERSON (<u>AusweisNr.</u>, Name, Vorname, PLZ, GebDatum)
ORT (<u>PLZ</u>, Ortsname)

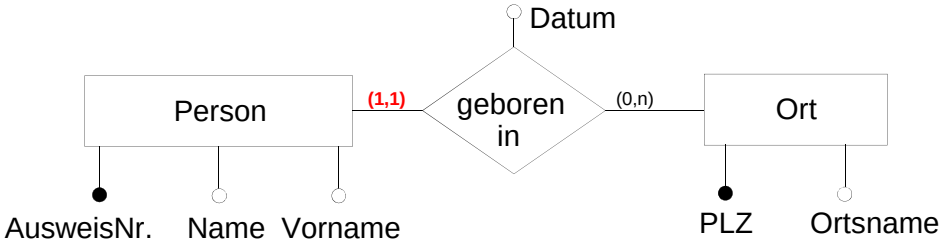
… aber auch möglich! (geht immer☺)



ENTITÄT_1(<u>Schlüssel1</u>, A_1)

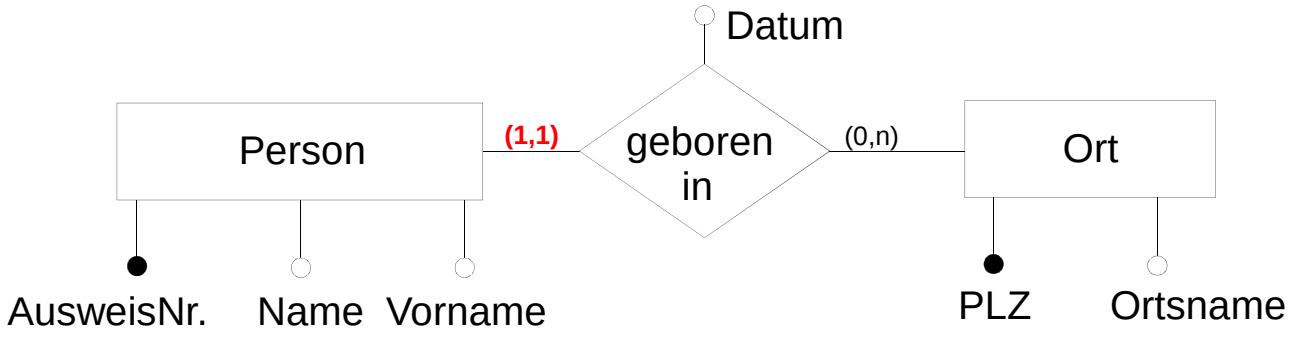
ENTITÄT_2 (<u>Schlüssel2</u>, A_2)

BEZIEHUNG (**<u>Schlüssel1</u>**, Schlüssel2, B)



PERSON (<u>AusweisNr.</u>, Name, Vorname)
ORT (<u>PLZ</u>, Ortsname)
GEBOREN_IN (<u>AusweisNr.</u>, PLZ, Datum)

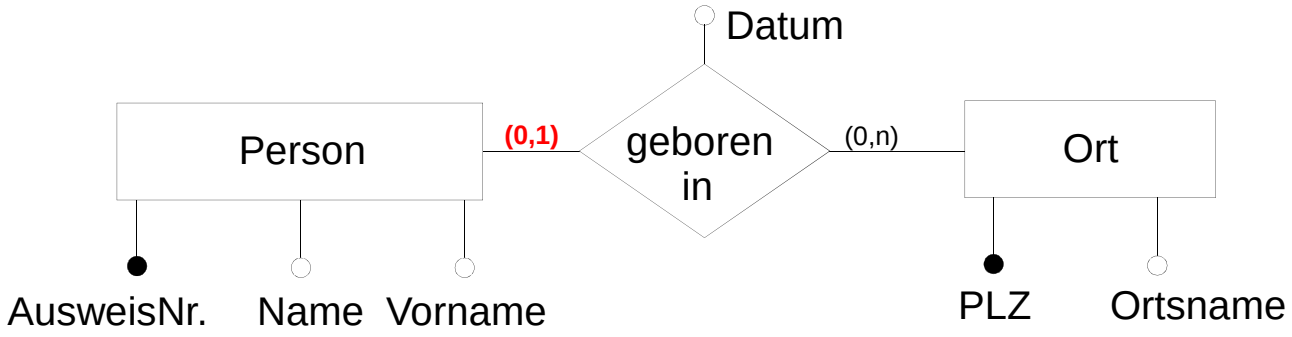
… Unterschied?



PERSON (<u>AusweisNr.</u>, Name, Vorname, PLZ, GebDatum)
ORT (<u>PLZ</u>, Ortsname)

PERSON (<u>AusweisNr.</u>, Name, Vorname)
ORT (<u>PLZ</u>, Ortsname)
GEBOREN_IN (<u>AusweisNr.</u>, PLZ, Datum)

… und hier?



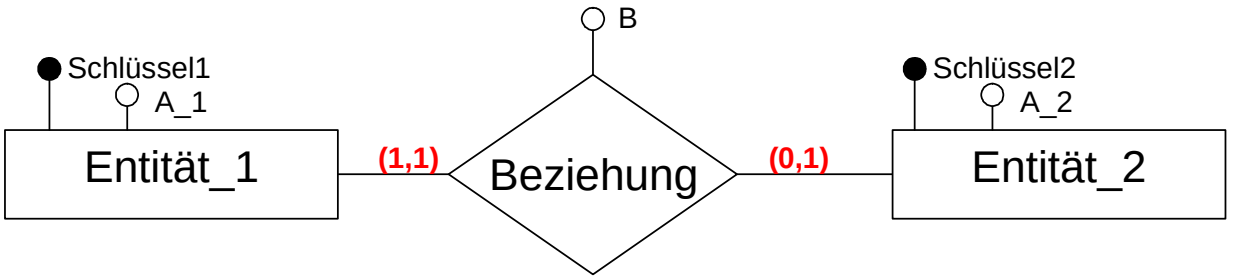
PERSON (<u>AusweisNr.</u>, Name, Vorname, PLZ, Datum)
ORT (<u>PLZ</u>, Ortsname)

PERSON (<u>AusweisNr.</u>, Name, Vorname)
ORT (<u>PLZ</u>, Ortsname)
GEBOREN_IN (<u>AusweisNr.</u>, PLZ, Datum)

(1,1) und (0,1)

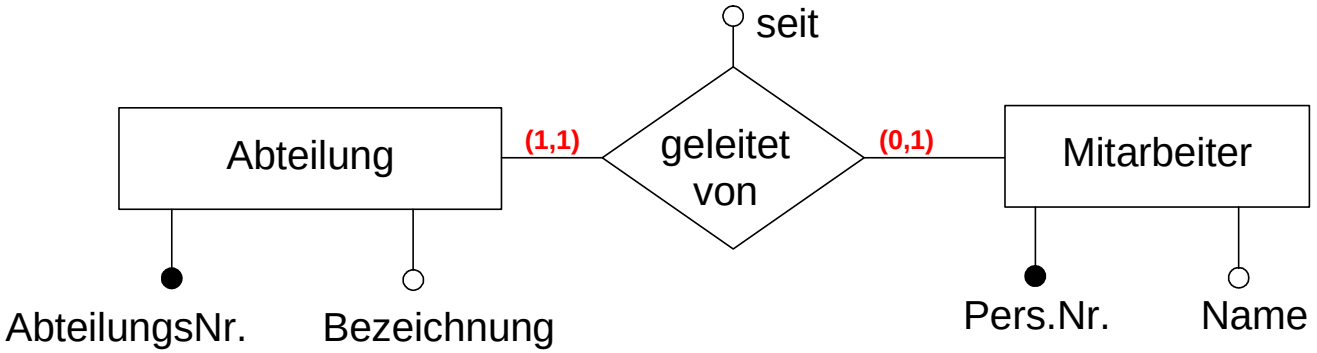
Sind beide min-Kardinalitäten = 0, so muss das allgemeine Verfahren ((0,n) und (0,n)) angewendet werden.

Ist nur eine min-Kardinalität = 1, so wendet man die Abbildung der one-to-many bzw. many-to-one Beziehung an.



ENTITÄT_1 (<u>Schlüssel1</u>, A_1, Schlüssel2, B)
ENTITÄT_2 (<u>Schlüssel2</u>, A_2)

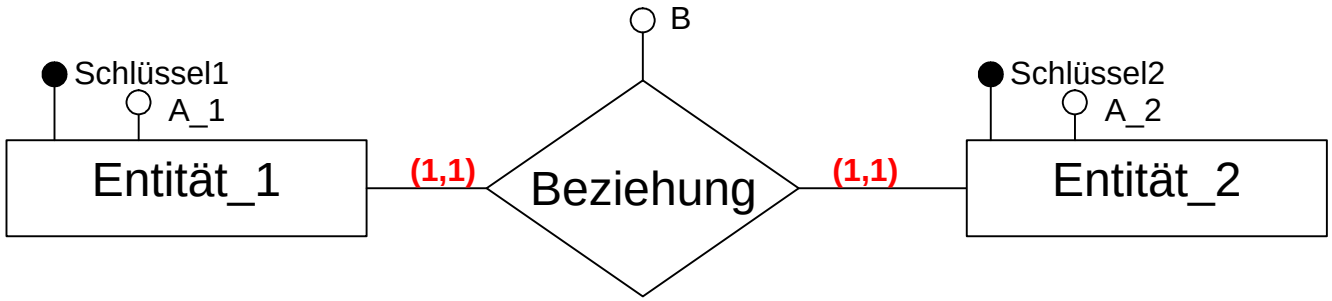
Beispiel



↓

ABTEILUNG (AbteilungsNr., Bezeichnung, Pers.Nr., seit)
MITARBEITER (Pers.Nr., Name)

(1,1) und (1,1)

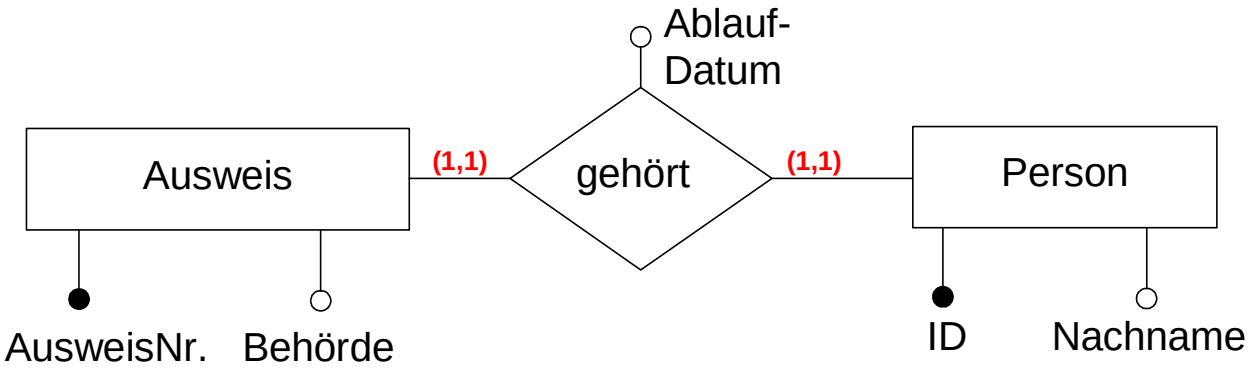


ENTITÄT_1_2 (Schlüssel1, A_1, Schlüssel2, A_2, B)

oder
ENTITÄT_1_2 (Schlüssel1, A_1, Schlüssel2, A_2, B)

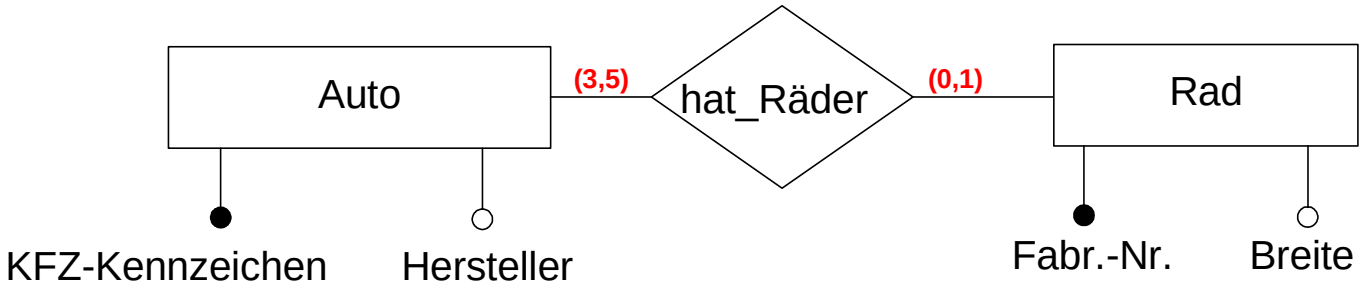
Nur noch eine Relation notwendig!

Beispiel



PERSON (<u>AusweisNr.</u>, Behörde, Ablaufdatum, ID, Nachname)

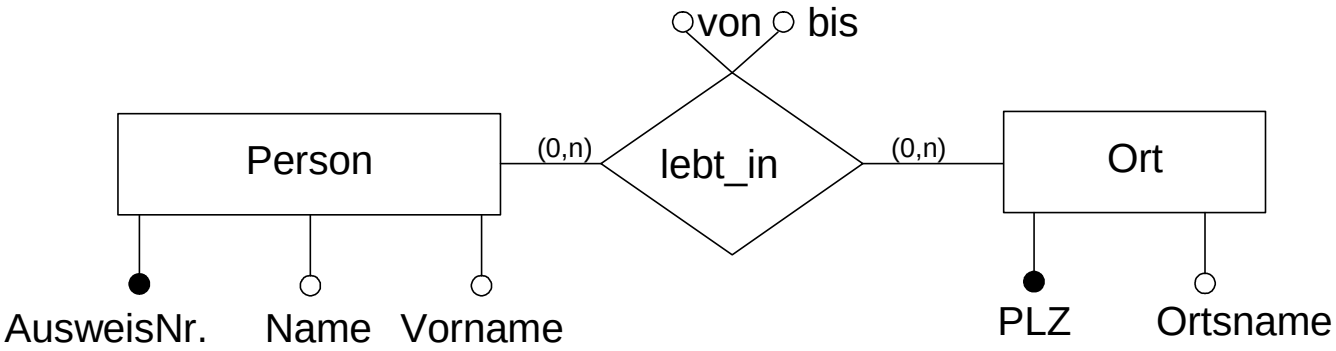
Sonderfälle



(Hier sind RAD1 – RAD3 verbindlich, also NOT NULL, während RAD4 und RAD5 durchaus Nullwerte beinhalten dürfen.)

AUTO (KFZ-Kennzeichen, Hersteller, RAD1, ... RAD5)
RAD (Fabr.-Nr., Breite)

Überführung in SQL



PERSON	(<u>AusweisNr.</u>, Name, Vorname)
ORT	(<u>PLZ</u>, Ortsname)
LEBT_IN	(<u>AusweisNr.</u>, <u>PLZ</u>, <u>von</u>, bis)

PERSON

AusweisNr	Name	Vorname
001	Jon	Lucas
003	Jon	Smith
103	Lucas	Jon

ORT

PLZ	Ortsname
501	Buli
503	Wali
603	Kali

LEBT_IN

AusweisNr	PLZ	von	bis
001	501	23.12.2000	25.12.2010
003	501	17.08.2004	Null
001	503	01.01.1999	01.01.2012
001	501	01.01.2012	Null

Fragen:

1. Welche Datentypen für die Attribute?
2. Welche Attribute dürfen Null-Values enthalten?
3. Wie erzeuge ich die referenzielle Integrität?

ORT

PLZ	Ortsname
501	Buli
503	Wali
603	Kali

z.B. varchar(40) oder besser varchar(45)? …

LEBT_IN

AusweisNr	PLZ	von	bis
001	501	23.12.2000	25.12.2010
003	501	17.08.2004	Null
001	503	01.01.1999	01.01.2012
001	501	01.01.2012	Null

Ist auch „01“ ein gültiger Wert?

Llanfairpwllgwyngyllgogerychwyrndrobwllllantysiliogogogoch ist ein walisischer Ortsname mit 58 Zeichen.

“Die *längste* Ortsbezeichnung[9] besitzt ein neuseeländischer Hügel namens *Taumatawhakatangihangakoauauotamateaturipukakapikimaungahoronukupokaiwhenuakitanatahu* (83 Zeichen), überboten vom offiziell verständlicherweise selten verwendeten Namen von Bangkok, *Krung Thep Mahanakhon Amon Rattanakosin Mahinthara Ayuthaya Mahadilok Phop Noppharat Ratchathani Burirom Udomratchaniwet Mahasathan Amon Piman Awatan Sathit Sakkathattiya Witsanukam Prasit* (168 Zeichen ohne Leerzeichen, 21 Wörter).[10] Europas längsten Ortsnamen trägt die walisische Ortschaft *Llanfairpwllgwyngyllgogerychwyrndrobwllllantysiliogogogoch* (58 Zeichen), wobei diese eine Städtepartnerschaft mit dem niederländischen Dorf Ee und dem französischen Dorf Y einging.”

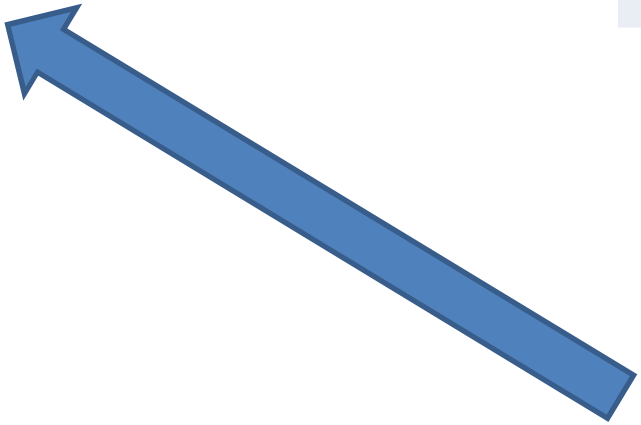
Aus: https://de.wikipedia.org/wiki/Ortsname#Lange_und_kurze_Ortsnamen

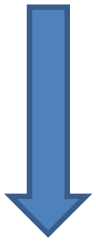
ORT

PLZ	Ortsname
501	Buli
503	Wali
603	Kali

LEBT_IN

AusweisNr	PLZ	von	bis
001	501	23.12.2000	25.12.2010
003	501	17.08.2004	Null
001	503	01.01.1999	01.01.2012
001	501	01.01.2012	Null





Werte hier sollen eine Referenz auf Objekte darstellen. Sie sollten enthalten sein in der Tabelle ORT!

Solche Verweise auf Schlüssel anderer Tabellen nennt man „Fremdschlüssel“!

ORT

PLZ	Ortsname
501	Buli
503	Wali
603	Kali

LEBT_IN

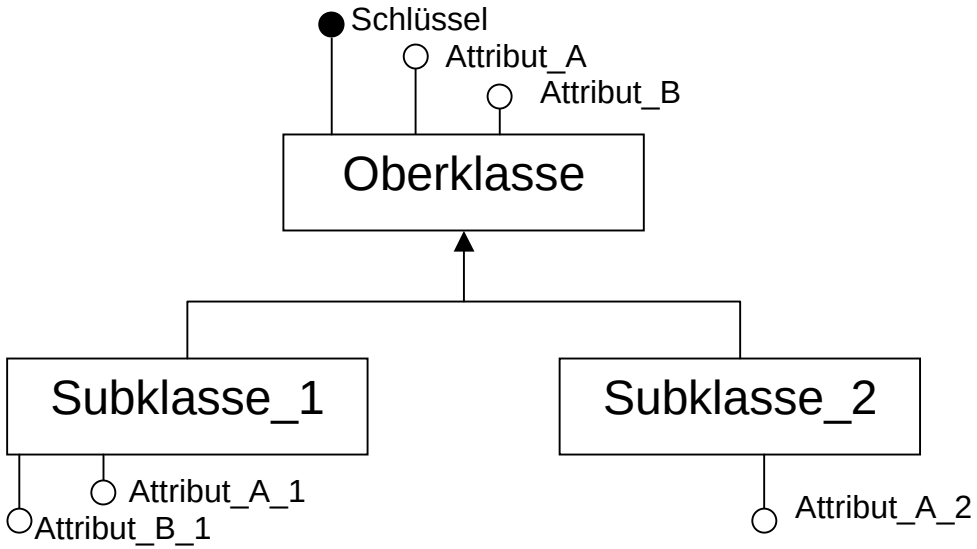
AusweisNr	PLZ	von	bis
001	501	23.12.2000	25.12.2010
003	501	17.08.2004	Null
001	503	01.01.1999	01.01.2012
001	501	01.01.2012	Null

```
CREATE  TABLE lebt_in (
  AusweisNr INT NOT NULL ,
  PLZ INT NOT NULL ,
  von DATE NOT NULL ,
  bis DATE NULL ,
  PRIMARY KEY (AusweisNr, PLZ, von) ,
    FOREIGN KEY (PLZ) REFERENCES ort (PLZ),
    FOREIGN KEY (AusweisNr) REFERENCES person (AusweisNr));
```

Definition

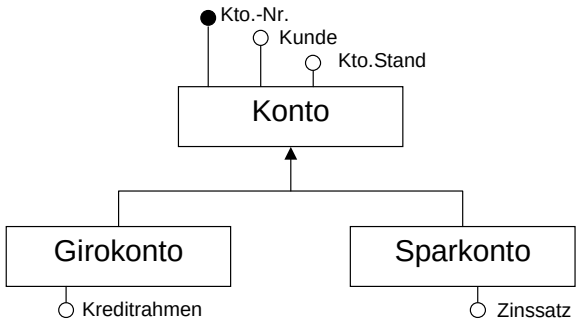
Referenzielle Integrität bedeutet die Konsistenz zwischen verbundenen Tabellen. *Referenzielle Integrität* wird durch die Kombination von Primärschlüssel und Fremdschlüssel erzwungen. Um die *referenzielle Integrität* zu erhalten darf jedes Feld einer Tabelle, das als Fremdschlüssel deklariert worden ist, nur Werte des entsprechenden Primärschlüssels der „Eltern Tabelle“ annehmen.

Abbildung der Generalisierung



Es gibt unterschiedliche Möglichkeiten dies ins rel. Modell abzubilden.

Generalisierung ER → rel. Modell



Möglichkeit 1:

KONTO (Kto.Nr., Kunde, Kto.Stand)
GIROKONTO (Kto.Nr., Kunde, Kto.Stand, Kreditrahmen)
SPARKONTO (Kto.Nr., Kunde, Kto.Stand, Zinssatz)

Verknüpfung zwischen Kto.Nr. über Fremdschlüssel!

Möglichkeit 2:

KONTO (Kto.Nr., Kunde, Kto.Stand)
GIROKONTO (Kto.Nr., Kreditrahmen)
SPARKONTO (Kto.Nr., Zinssatz)

Möglichkeit 3:

KONTO (Kto.Nr., Kunde, Kto.Stand, Kreditrahmen, Zinssatz)

manchmal auch mit Type-Spalten (diese enthalten einfach true/false Info):
KONTO (Kto.Nr., Kunde, Kto.Stand, **Giro, Spar**, Kreditrahmen, Zinssatz)

nicht klausurrelevant☺

Zusammenfassung

- Übersetzung ER-Modell ins Relationenmodell (nicht eindeutig!)
- Referenzielle Integrität mit Fremdschlüssel

Wieder verbinden in Anfragen?

PERSON

AusweisNr	Name	Vorname
001	Jon	Lucas
003	Jon	Smith
103	Lucas	Jon

ORT

PLZ	Ortsname
501	Buli
503	Wali
603	Kali

LEBT_IN

AusweisNr	PLZ	von	bis
001	501	23.12.2000	25.12.2010
003	501	17.08.2004	Null
001	503	01.01.1999	01.01.2012
001	501	01.01.2012	Null

Welche Personen leben aktuell in Buli?

```
select      A1,A2, ... ,An
from        R1, R2, ... ,Rm
[where      conditions]
[group by   clause]
[having     clause]
[order by   clause];
```

Wieder verbinden in Anfragen?

PERSON

AusweisNr	Name	Vorname
001	Jon	Lucas
003	Jon	Smith
103	Lucas	Jon

ORT

PLZ	Ortsname
501	Buli
503	Wali
603	Kali

LEBT_IN

AusweisNr	PLZ	von	bis
001	501	23.12.2000	25.12.2010
003	501	17.08.2004	Null
001	503	01.01.1999	01.01.2012
001	501	01.01.2012	Null

Welche Personen leben aktuell in Buli?

```
SELECT p.* FROM lebt_in l, person p, ort o
        where
                o.Ortsname = 'Buli' and
                o.plz = l.plz and
                p.ausweisnr = l.ausweisnr and
                l.bis is null;
```

Wieder verbinden in Anfragen?

PERSON

AusweisNr	Name	Vorname
001	Jon	Lucas
003	Jon	Smith
103	Lucas	Jon

ORT

PLZ	Ortsname
501	Buli
503	Wali
603	Kali

LEBT_IN

AusweisNr	PLZ	von	bis
001	501	23.12.2000	25.12.2010
003	501	17.08.2004	Null
001	503	01.01.1999	01.01.2012
001	501	01.01.2012	Null

Welche Personen leben aktuell in Buli?

```
SELECT p.* FROM lebt_in l
        join person p on (l.ausweisnr=p.ausweisnr)
        join ort o on (o.plz=l.plz)
        where
                o.Ortsname = 'Buli' and l.bis is null;
```

Wichtige DBMS 2007

- Oracle
- IBM DB2
- Microsoft SQL Serve
- …

Aber was heißt das eigentlich?

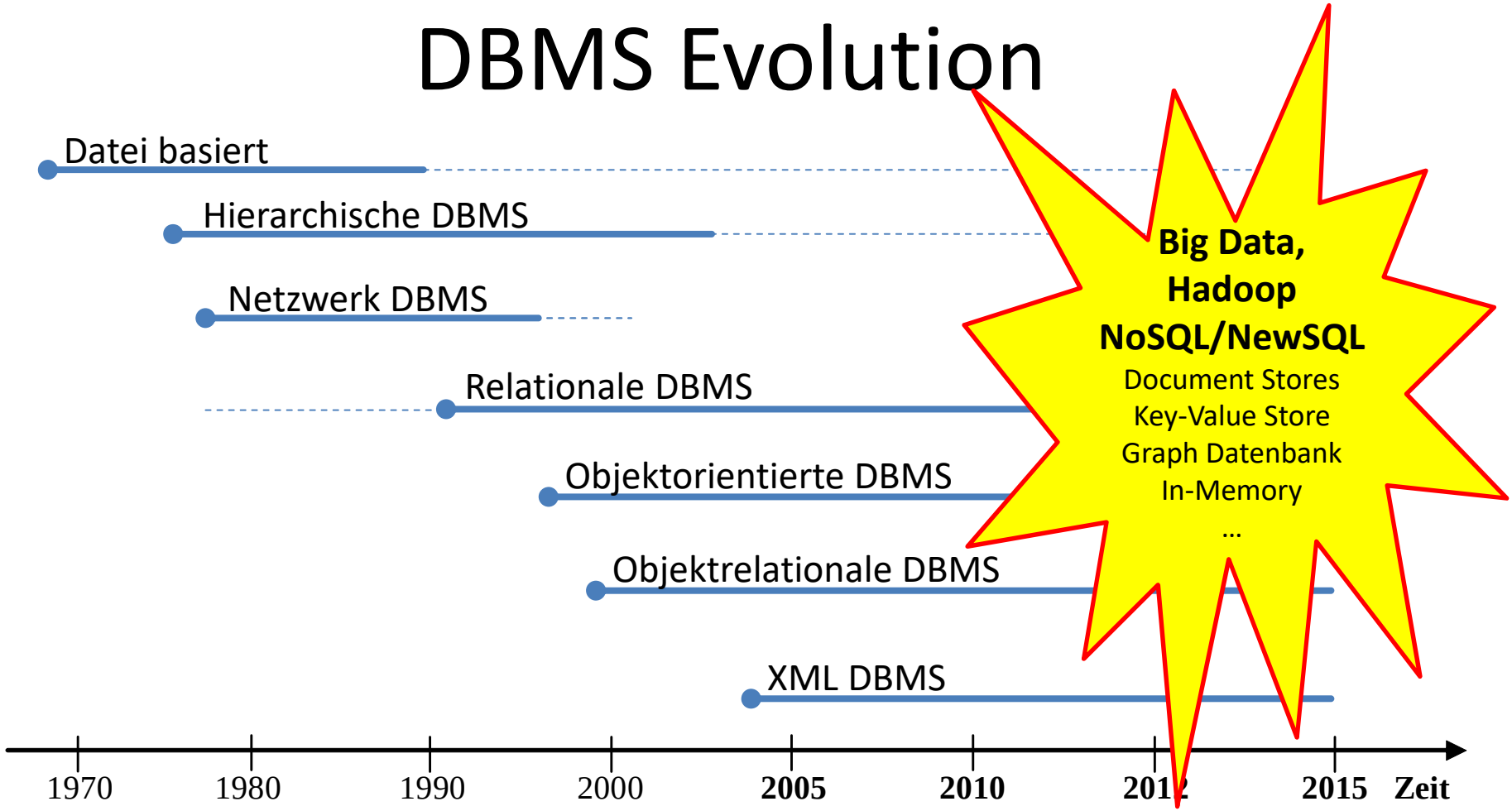
mpany	Revenue 2007	Market share 2007
Oracle	8,343 Mrd. Dollar	37,6%
IBM	4,879 Mrd. Dollar	22,0%
Microsoft	4,670 Mrd. Dollar	21,0%

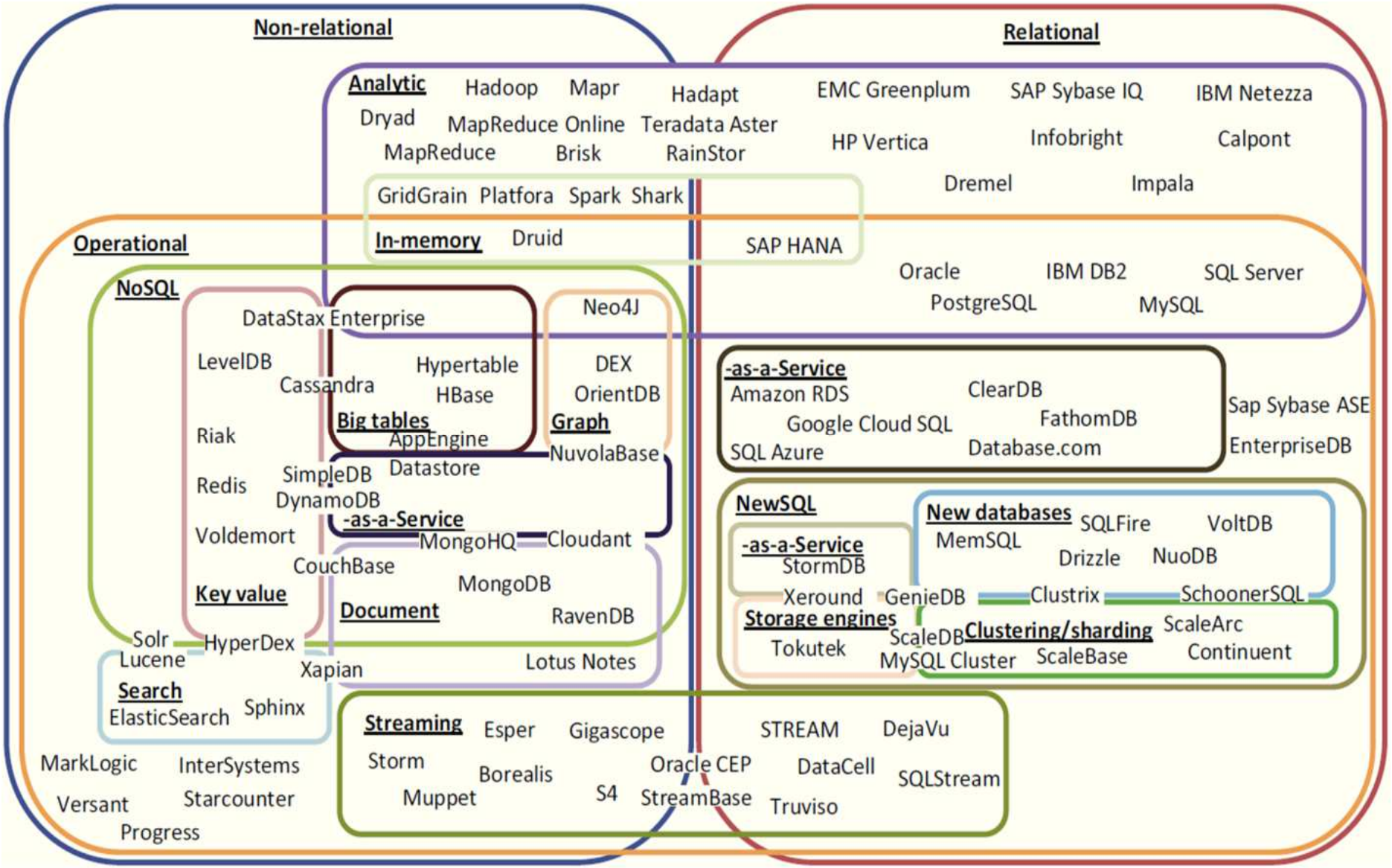
aus Computerwoche Nr. 3 vom 16. Januar 2009
Zahlen beziehen sich nur auf DBMS-Geschäft

Top 10 Enterprise Database Systems to Consider in 2015

http://www.serverwatch.com/server-trends/Top-10-Enterprise-Database-Systems-in-2015.html

DBMS Evolution





Reference: H. Lim, Y. Han, and S. Babu, “How to Fit when No One Size Fits.,” in *CIDR*, 2013.

Geschätzte Kosten für Ausfallzeiten (Downtimes) Stand: 2002!!!!

Geschäftsfeld	Geschätzte Kosten pro Stunde
Aktienhandel	
Kreditkarten-Gesellschaften	
Home Shopping Channel	$113.750
Katalog Verkaufs Center	$90.000
Fluglinien Reservierung	$89.500
Versand Service	$28.250
Geldautomat Service	$14.500

Nach: Craig S. Mullins
Database Administration – The Complete Guide to Practices and Procedures
2002 – Addison-Wesley – ISBN 0-201-74129-6

Kosten für Ausfallzeiten (Downtimes)

Ausfallzeiten pro Jahr			
Verfügbarkeit	Minuten	Stunden	Kosten pro Jahr*
99,999%	5	0,08	8.000 $
99,99%	53	0,88	88.000 $
99,9%	526	8,77	887.000 $
99,5%	2628	43,8	4.380.000 $
99%	5256	87,6	8.760.000 $

* Bei Kosten für eine Stunde Ausfallzeit von 100.000 $

Nach: Craig S. Mullins
Database Administration – The Complete Guide to Practices and Procedures
2002 – Addison-Wesley – ISBN 0-201-74129-6