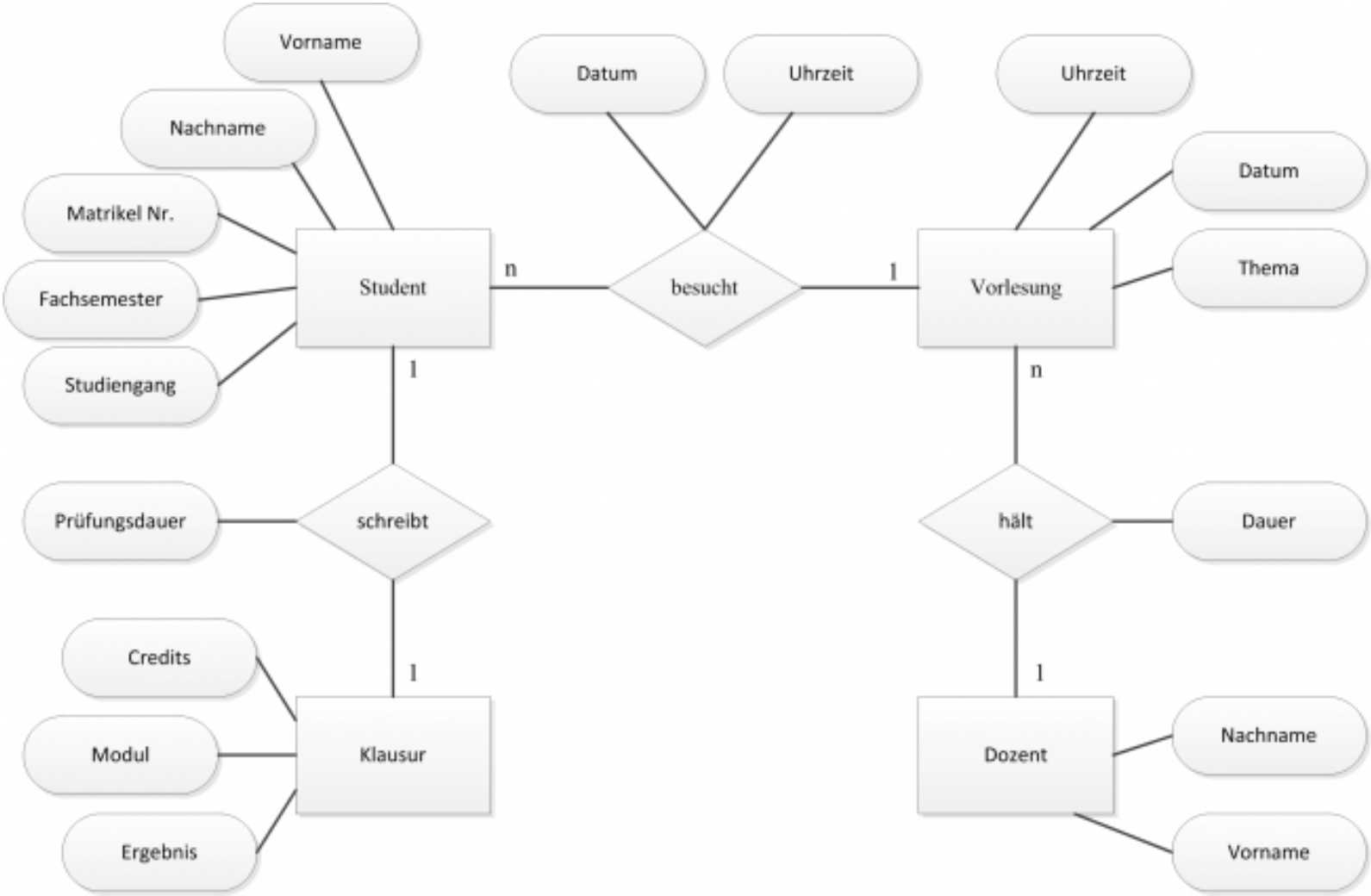
Datenmodellierung

TSBE Bern
Höhere Fachschule



Agenda

TSBE Bern
Höhere Fachschule

1. Ausgangslage
2. Was sind Datenmodelle?
3. Entity Relationship Diagram
4. Entity Relationship Model
5. Einsatzgebiete
6. Terminologie
7. Notation
8. Übung Webshop

Ausgangslage

TSBE Bern
Höhere Fachschule

- Die Daten eines Betriebs sind langlebig und wertvoll
- Oft sind sie das Herz einer Firma
- In der IT ändert sich alles, die Daten bleiben konstant
- “Im Kern geht es um Daten”
- Merkel: “Daten sind die Rohstoffe des 21. Jahrhunderts”

Was sine Datenmodelle?

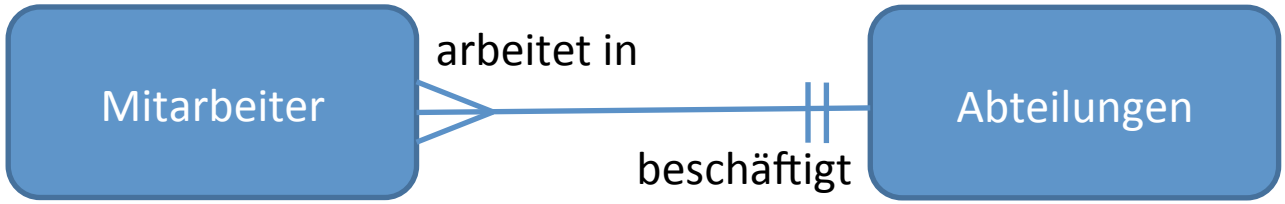
TSBE Bern
Höhere Fachschule

- Ein Datenmodell ist ein “Entity Relationship Model”, ERM
- Ein Datenmodell beinhaltet eine grafische Darstellung “Entity Relation Diagram”, ERD
- Es bildet Tabellen (Entitäten) und deren Beziehungen (Relationen) zu anderen Tabellen ab
- Ein Datenmodell ist DBMS unabhängig
- Es beschreibt das “was”, nicht das “wie”
- Das Ziel eines relationalen Datenmodells ist es möglichst keine redundanten Daten zu haben

Entiy Relationship Diagram

TSBE Bern
Höhere Fachschule

- Ein ERD ist die Basis jedes Datenmodells
- Das ERD ist die grafische Darstellung eines Datenmodells
- Es bildet die Beziehungen der Entitäten untereinander ab



Entity Relationship Model

TSBE Bern
Höhere Fachschule

- Ein ERM beinhaltet ein ERD (grafische Darstellung) sowie die darin enthaltenen Elemente und deren Bedeutung und Beziehung zueinander
- Ein ERM wird in der konzeptionellen Phase der SW Entwicklung verwendet. Es dient als Verständigung zwischen Anwender und Entwickler
- In der Implementierungsphase dient es als Grundlage für das Datenbankdesign
- ERM's sind quasi der Standard zur Datenmodellierung
- Es gibt unterschiedliche Darstellungsformen von ERM's

Einsatzgebiete

TSBE Bern
Höhere Fachschule

- Refactoring
  - Verstehen eines bestehenden Datenmodells
  - Erweitern eines bestehenden Datenmodells
- Grundkonzept
  - Strukturierter Neuaufbau eines Datenmodells
- Logische Organisation der Daten
  - Konzeptionelles Schema
- Physische Organisation der Daten
  - Internes Schema
- Beschreiben der Datenarchitektur eines Systems

Terminologie

TSBE Bern
Höhere Fachschule

- Entity, Entität, Entitätsmenge, Tupel, Row, Datensatz
- Beziehung, Relation
- Konzeptionelles Modell, Logisches Modell
- Internes Modell, Physisches Modell
- Beschreiben der Datenarchitektur eines Systems
- Attribut

Entität

TSBE Bern
Höhere Fachschule

- Eine Entität ist ein in sich geschlossenes Objekt
- Eine Entität ist eindeutig identifizierbar
- Synonyme: Datensatz, Row, Record, Tupel
- In der Datenmodellierung wird mit Entitätsmengen gearbeitet
- Achtung: Oft wird an Stelle von Entitätsmengen von Entitäten gesprochen

Entitätsmengen

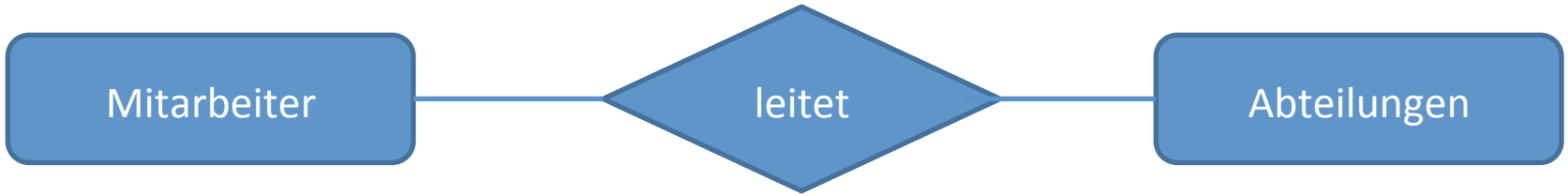
TSBE Bern
Höhere Fachschule

- Eine Entitätsmenge ist eine Menge mehrerer gleicher Entitäten
- Eine Entitätsmenge entspricht einer Tabelle in der Datenbank
- Best Practices Namensgebung: Eine Entitätsmenge beinhaltet mehrere gleiche Entitäten: → Bezeichnung in Mehrzahl verwenden

Beziehungen

TSBE Bern
Höhere Fachschule

- Beziehungen = Relations
- Entitäten stehen in Beziehung zu anderen Entitäten



- Beziehungen zwischen Entitäten sollten immer sprechend, in beide Richtungen, bezeichnet werden
  - «Mitarbeiter» «leitet» «Abteilung»
  - «Abteilung» «wird geleitet von» «Mitarbeiter»

Attribut

TSBE Bern
Höhere Fachschule

- Eine Entität besteht aus mehreren Attributen
- Die Attribute einer Entität bestimmen ihre Eigenschaften
  - Entität Personen:
  - Attribute:
    - Name
    - Vorname
    - Adresse
    - PLZ
    - Ort

Kardinalität

TSBE Bern
Höhere Fachschule

- Die Kardinalität der Beziehungen zwischen Entitäten sind unterschiedlich
  - Eine Person wohnt in genau einem Ort
  - In einem Ort wohnen mehrere Personen
- Es gibt folgende Möglichkeiten:
  - 1:1
  - 1:0/1
  - 1:n
  - 1:0/n
- m:n Abbildungen sind nicht möglich, sie können nicht abgebildet werden
  - m:n Abbildungen werden durch eine Zwischentabelle aufgelöst

1:1 Beziehungen

TSBE Bern
Höhere Fachschule

- Bei einer «eins zu eins» Beziehung in einer relationalen Datenbank ist jede Row der Tabelle A genau einer Row in der Tabelle B- oder umgekehrt- zugeordnet
- Bei 1:1 Verbindungen muss man sich überlegen, ob es nicht Sinn macht alle Attribute in eine Tabelle zu schreiben
- Gründe um mehrere Tabellen zu machen könnten folgende sein:
  - Unterschiedliche Zugriffsrechte auf beide Tabellen
  - Reduzieren der Komplexität
- Beispiel:
  - Jede Home Cinema Anlage hat genau einen Subwoofer

1:0/1 Beziehungen

TSBE Bern
Höhere Fachschule

- Bei einer «eins zu Null oder eins» Beziehung in einer relationalen Datenbank ist jede Row von Tabelle A mit genau einer- oder keiner Row in Tabelle B verbunden
- Beispiel:
  - Jede Person hat einen- oder keinen Ehepartner

1:n Beziehungen

TSBE Bern
Höhere Fachschule

- Bei einer «eins zu viele» Beziehung in einer relationalen Datenbank ist jede Row von Tabelle A mit einer- oder mehreren Rows Tabelle B verbunden
- Jede Row in Tabelle B ist jedoch nur mit genau einer Row in Tabelle A verbunden
- Beispiel:
  - Jeder Lehrer unterrichtet ein- oder mehrere Fächer

1:0/n Beziehungen

TSBE Bern
Höhere Fachschule

- Bei einer «eins zu Null oder viele» Beziehung in einer relationalen Datenbank ist jede Row von Tabelle A mit keiner- oder mehreren Rows Tabelle B verbunden
- Jede Row in Tabelle B ist jedoch nur mit genau einer Row in Tabelle A verbunden
- Beispiel:
  - Jeder Kunde hat keine-, eine- oder mehrere Rechnungen

m:n Beziehungen

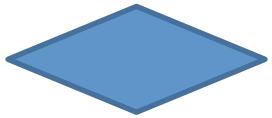
TSBE Bern
Höhere Fachschule

- «Viele zu viele» Beziehungen können in einem relationalen Datenmodell nicht abgebildet werden. Sie müssen durch eine Zwischentabelle in zwei 1:n – n:1 Beziehungen aufgelöst werden
- Beispiel
  - Jede Rechnung enthält mehrere Artikel, jeder Artikel kann auf mehreren Rechnungen erscheinen

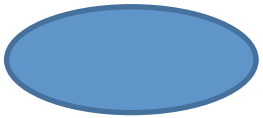
Notation

TSBE Bern
Höhere Fachschule

- Entity, Entitätsmengen



- Beziehung



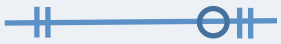
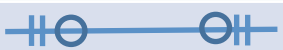
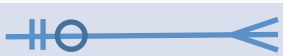
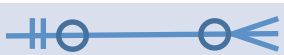
- Attribut

- Schlüsselattribut

Notation Kardinalität

TSBE Bern
Höhere Fachschule

- Es gibt viele verschiedene Möglichkeiten der Notation von Beziehungen
- Weit verbreitet ist die «Krähenfuss» Notation
- Auch bei dieser Notation gibt es kleine Unterschiede, wenn man sie jedoch kennt, ist klar was gemeint ist
- Wichtig sind ist die Bedeutung der «Besen» und der «Nullen»

	1	1 oder 0	n	n oder 0
1				
1 oder 0				
n				
n oder 0				

Unterschiedliche Notationen

TSBE Bern
Höhere Fachschule

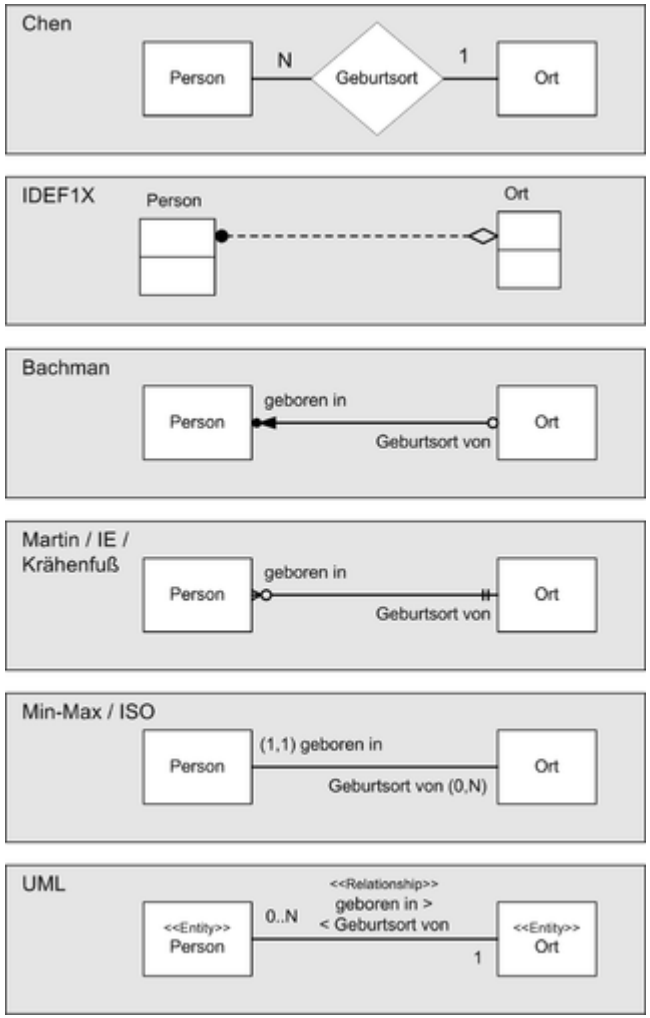


Bild: Wikipedia

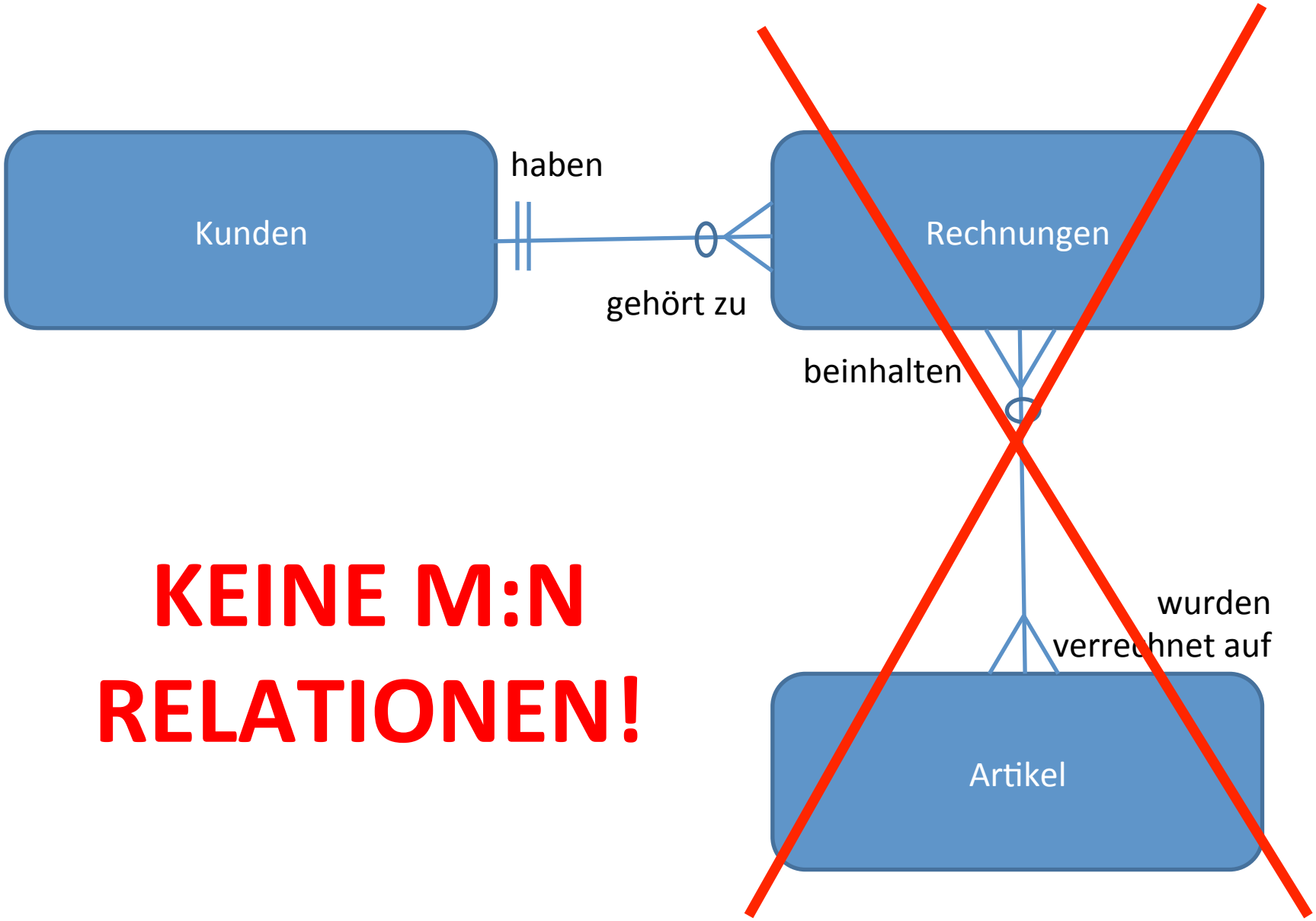
Übung: ERD Webshop Teil 1

TSBE Bern
Höhere Fachschule

- Beispiel eines einfachen Webshop’s
- Welche Entitäten sind nötig um eine Webshop in einer Datenbank abbilden zu können?
  - Kunden
  - Artikel
  - Rechnungen

Übung: ERD Webshop Teil 1

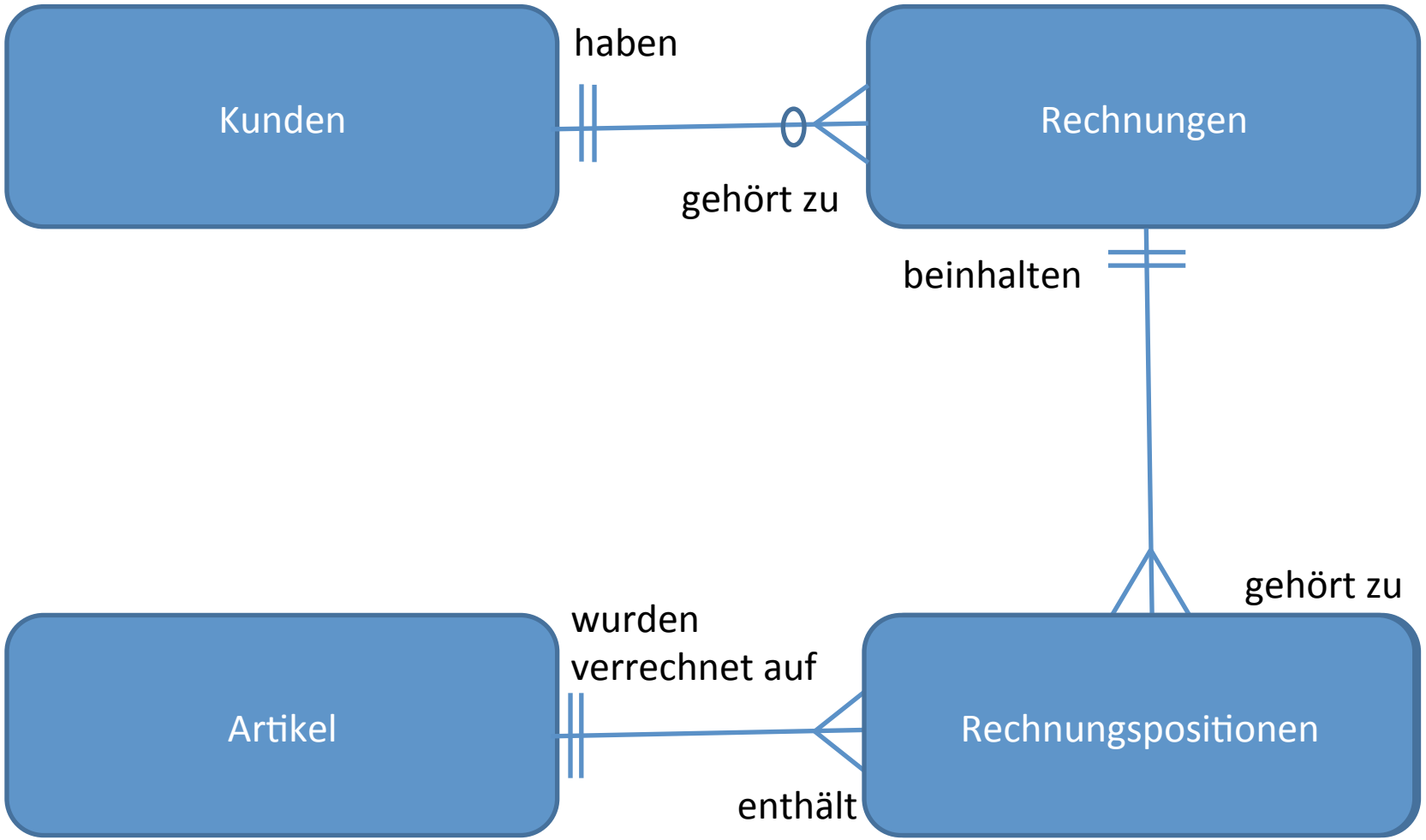
TSBE Bern
Höhere Fachschule



KEINE M:N RELATIONEN!

Übung: ERD Webshop Teil 1

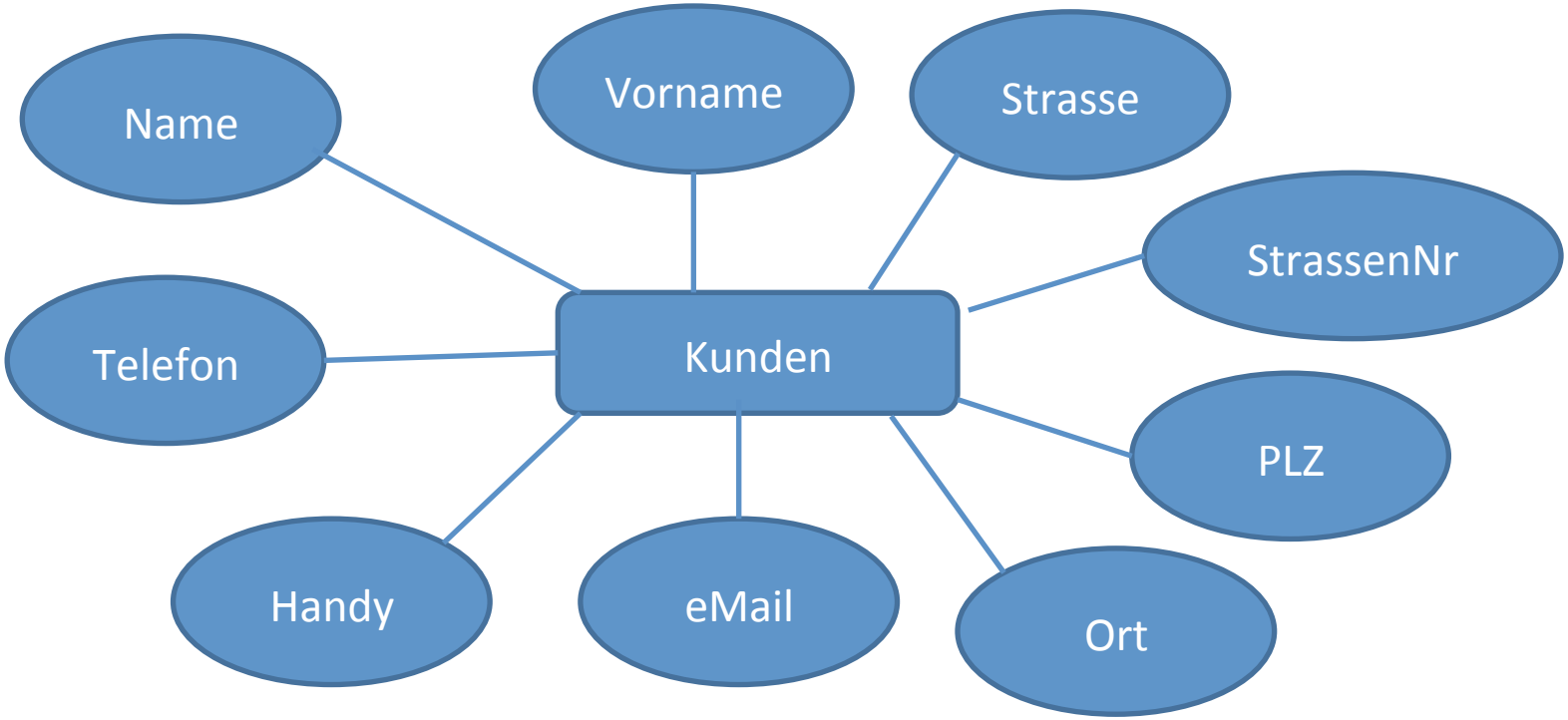
TSBE Bern
Höhere Fachschule



Übung: Tabelle Kunden

TSBE Bern
Höhere Fachschule

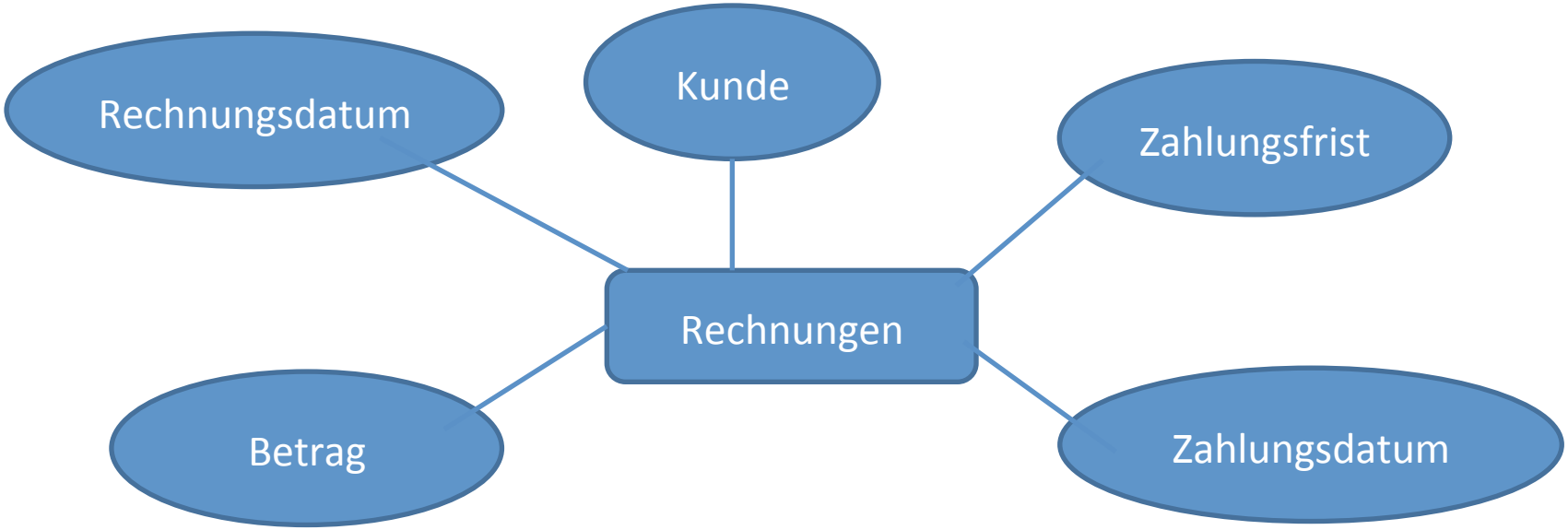
- Welche Attribute braucht die Tabelle «Kunden?»
  - Welche Informationen will ich darin abspeichern?



Übung: Tabelle Rechnungen

TSBE Bern
Höhere Fachschule

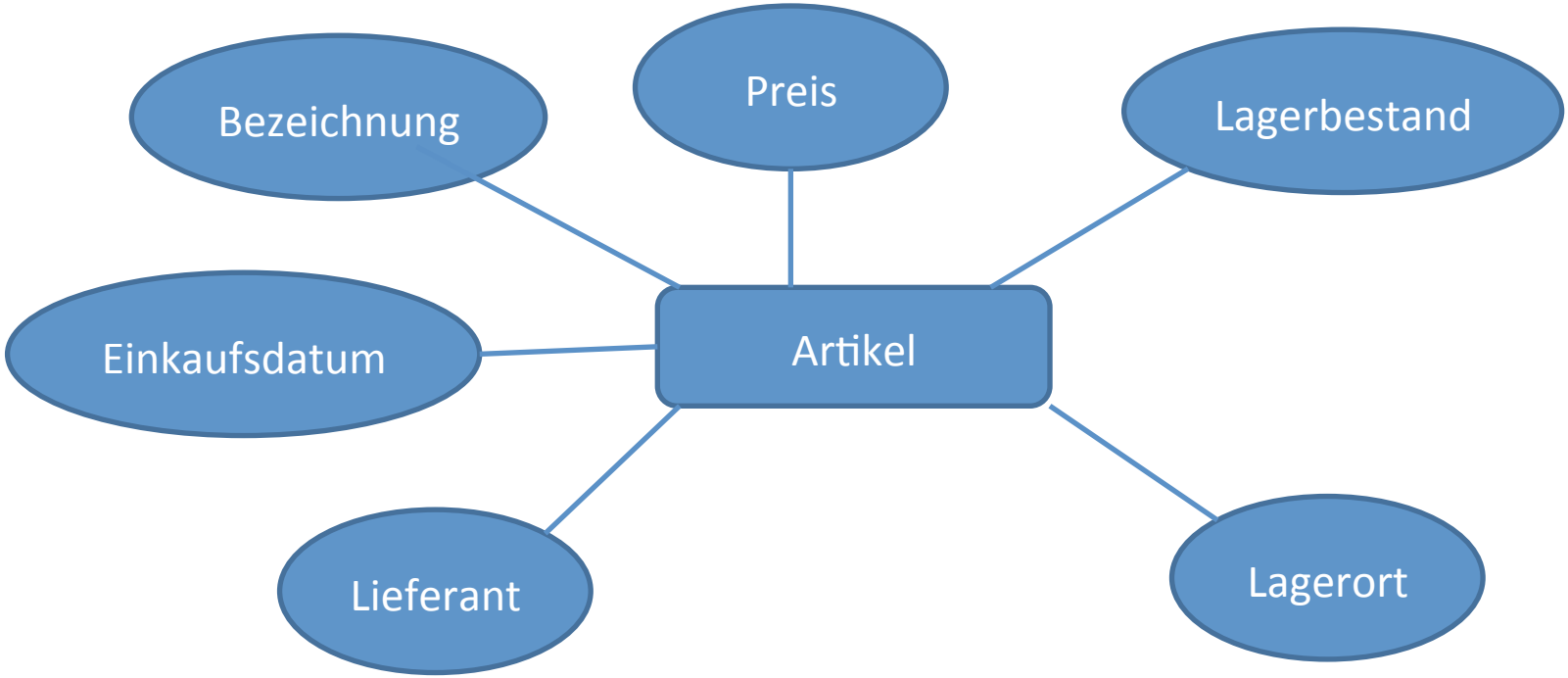
- Welche Attribute braucht die Tabelle «Rechnungen?»



Übung: Tabelle Artikel

TSBE Bern
Höhere Fachschule

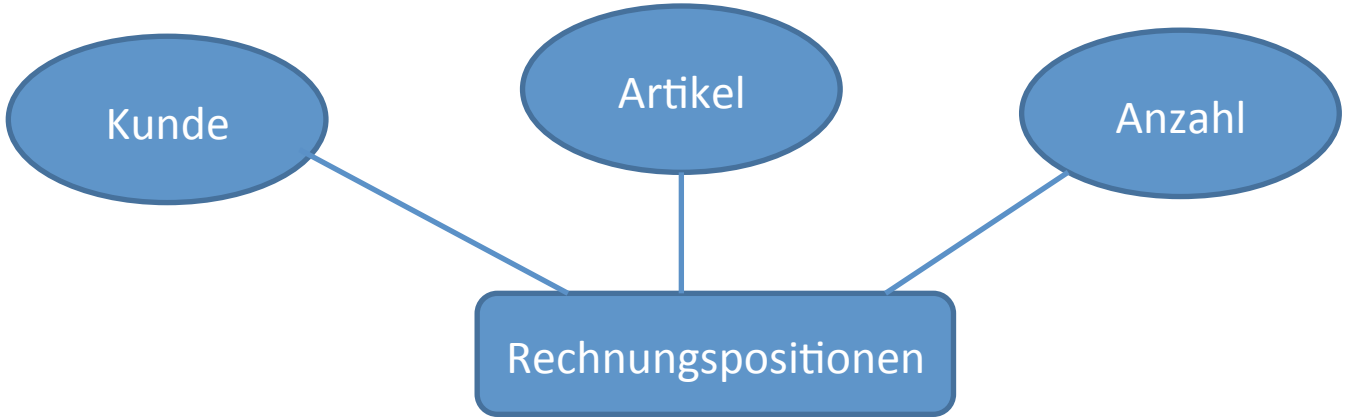
- Welche Attribute braucht die Tabelle «Artikel?»



Übung: Tabelle RechnungsPositionen

TSBE Bern
Höhere Fachschule

- Welche Attribute braucht die Tabelle «Rechnungspositionen?»



Übung: Tabelle Kunden

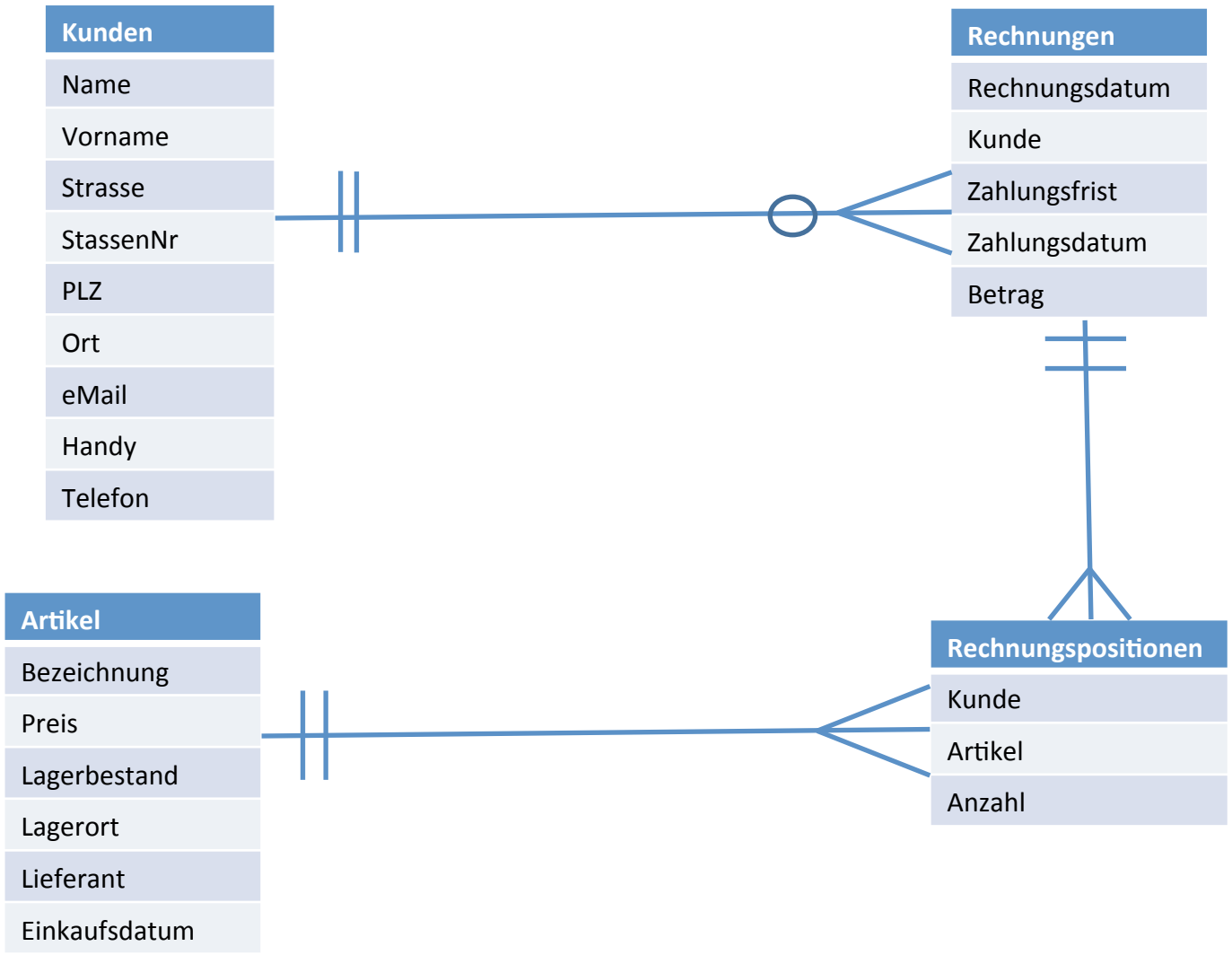
TSBE Bern
Höhere Fachschule

- Andere Notation:
  - Braucht etwas weniger Platz
  - Ist mE nach übersichtlicher

Kunden
Name
Vorname
Strasse
StrassenNr
PLZ
Ort
eMail
Handy
Telefon

Übung: Webshop

TSBE Bern
Höhere Fachschule



Physisches Modell

TSBE Bern
Höhere Fachschule

- Um das Datenmodell in der Datenbank abbilden zu können, müssen Tabellen erstellt werden
- Tabellen sind das physische Abbild einer Entitätsmenge
- Tabellen bestehen aus mehreren Attributen mit eigenen Datentypen
- Gängige Datentypen:
  - NUMBER
  - CHAR
  - VARCHAR2
  - DATE
  - TIMESTAMP
  - CLOB (Character Large Object)
  - BLOB (Binary Large Object)

Datentyp NUMBER

TSBE Bern
Höhere Fachschule

- Im Datentyp NUMBER werden Zahlen abgespeichert
- Positive Zahlen zwischen 1×10^{-130} und $9.99...9 \times 10^{125}$ mit bis zu 38 unterschiedlichen Zahlen
- Negative Zahlen zwischen -1×10^{-130} und $9.99...99 \times 10^{125}$ mit bis zu 38 unterschiedlichen Zahlen
- NUMBER kann eingeschränkt werden um Platz zu sparen:
  - NUMBER(2) → zwei Stellen (-99 bis 99)
  - NUMBER(3,2) → total drei Stellen mit zwei Nachkommastellen (-9.99 bis 9.99)

Datentyp CHAR 1/2

TSBE Bern
Höhere Fachschule

- Im Datentyp CHAR werden alphanumerische Zeichen abgespeichert
- CHAR alloziert immer genau den Platz, der definiert wurde (siehe später VARCHAR)
  - Ist der String kürzer, wird er mit Leerschlägen aufgefüllt
- CHAR wird nicht mehr so oft verwendet weil es meist zu viel Platz braucht
- Möglicher Einsatzbereich:
  - Boolesche Werte (T/F)
  - Attribute, die immer mit denau dieser Anzahl Zeichen gefüllt werden

Datentyp CHAR 2/2

TSBE Bern
Höhere Fachschule

```
SQL> create table test (a char(20));

Table created.

SQL> insert into test(a) values('Hallo');

1 row created.

SQL> select '.' || a || '.' from test;

'.'||A||'.'
----------------------
.Hallo               .

SQL> select '.' || rtrim(a) || '.' from test;

'.'||RTRIM(A)||'.'
----------------------
.Hallo.
```

Datentyp VARCHAR

TSBE Bern
Höhere Fachschule

- Bei ORACLE heisst der Datentyp VARCHAR2, nicht VARCHAR. (warum auch immer…)
- Im Datentyp VARCHAR2 werden alphanumerische Zeichen abgespeichert
- VARCHAR2 alloziert nur den Platz, der auch verwendet wird
- Um Strings abzuspeichern wird meistens VARCHAR2 verwendet

Datentyp DATE 1/2

TSBE Bern
Höhere Fachschule

- Im Datentyp DATE wird ein Datum abgespeichert
- Im Datentyp DATE werden folgende Informationen abgespeichert:
  - Tag
  - Monat
  - Jahr (4 Stellig)
  - Stunde
  - Minute
  - Sekunde

Datentyp DATE 2/2

TSBE Bern
Höhere Fachschule

- Die Ausgabe von DATE kann angepasst werden

```
SQL> select sysdate from dual;

SYSDATE
---------
29-FEB-16

SQL> alter session set nls_date_format='dd.mm.yyyy hh24:mi:ss';

Session altered.

SQL> select sysdate from dual;

SYSDATE
-------------------
29.02.2016 22:31:15

SQL> alter session set nls_date_format='day.Mon.yyyy';

Session altered.

SQL> select sysdate from dual;

SYSDATE
------------------
monday     .Feb.2016
```

Datentyp TIMESTAMP

TSBE Bern
Höhere Fachschule

- Im Datentyp TIMESTAMP wird auch ein Datum abgespeichert
- TIMESTAMP ist präziser als DATE
- Im Datentyp TIMESTAMP werden folgende Informationen abgespeichert:
  - Tag
  - Monat
  - Jahr (4 Stellig)
  - Stunde
  - Minute
  - Sekunde
  - Sekundenbruchteile
  - Bei Bedarf auch die Zeitzone (TIMESTAMP WITH TIMEZONE)

```
SQL> select systimestamp from dual;

SYSTIMESTAMP
---------------------------------------------------------------------------
29-FEB-16 10.36.28.088089 PM +01:00
```

Physisches Modell Tabelle Kunden

TSBE Bern
Höhere Fachschule

Kunden	Datentyp
Name	varchar2(50)
Vorname	varchar2(50)
Adresse	varchar2(70)
PLZ	number(5)
Ort	varchar2(40)
eMail	varchar2(60)
Handy	varchar2(13)
Telefon	varchar2(13)

Physisches Modell Tabelle Artikel

TSBE Bern
Höhere Fachschule

Artikel	Datentyp
Bezeichnung	varchar2(50)
Preis	number(7,2)
Lagerbestand	number(3)
Lagerort	varchar2(50)
Lieferant	varchar2(40)
Einkaufsdatum	date

Physisches Modell Tabelle Rechnungen

TSBE Bern
Höhere Fachschule

Rechnungen	Datentyp
Rechnungsdatum	date
Kunde	?
Zahlungsfrist	number(2)
Zahlungsdatum	date
Betrag	Number(8,2)

Physisches Modell Tabelle Rechnunspos.

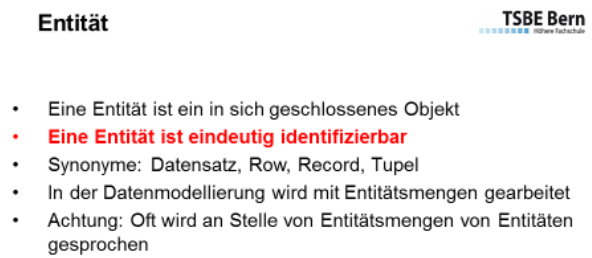
TSBE Bern
Höhere Fachschule

RechnungsPos	Datentyp
Rechnung	?
Artikel	?
Anzahl	number(4)

Eindeutiges Identifizieren 1/3

TSBE Bern
Höhere Fachschule

- Eine Entität muss eindeutig identifizierbar sein
- …siehe da ☺



- Wie kann das erreicht werden?

Eindeutiges Identifizieren 2/3

TSBE Bern
Höhere Fachschule

- Beispiel Tabelle Kunden:
- Wie kann man eine Row eindeutig identifizieren?
  - Name? → Schlecht. Und wenn es einen zweiten Müller gibt?
  - Name und Vorname? → Hmmm…
  - Name und Vorname und Geburtsdatum?
    - Wäre eine Möglichkeit, ist aber immer noch nicht garantiert eindeutig und unpraktisch dazu
- Lösung:
  - Eine Nummer

Eindeutiges Identifizieren 3/3

TSBE Bern
Höhere Fachschule

- Bei «Mastertabellen» wird oft eine Nummer als Primary Key (Primärschlüssel) angelegt
- «Mastertabellen» sind nicht abgeleitete Tabellen, keine Zwischentabellen wie in diesem Beispiel die Tabelle Rechnungspositionen
- In Zwischentabellen wird der Primary Key meist aus den Primary Key's zusammengesetzt, aus welchen die Tabelle abgeleitet ist
- Ein Primary Key identifiziert eine Row eindeutig. Er ist einmalig (unique) in einer Tabelle
- Jede Tabelle braucht immer und in jedem Fall einen Primary Key!

Übung: Webshop

TSBE Bern
Höhere Fachschule

Kunden	Datentyp
KundenNr	number(6)
Name	varchar2(50)
Vorname	varchar2(50)
Adresse	varchar2(70)
PLZ	number(5)
Ort	varchar2(40)
eMail	varchar2(60)
Handy	varchar2(13)
Telefon	varchar2(13)

Rechnungen	Datentyp
RechnungsNr	number(6)
Rechnungsdatum	date
KundenNr	number(6)
Zahlungsfrist	number(2)
Zahlungsdatum	date
Betrag	number(8,2)

Artikel	Datentyp
ArtikelNr	number(6)
Bezeichnung	varchar2(50)
Preis	number(7,2)
Lagerbestand	number(3)
Lagerort	varchar2(50)
Lieferant	varchar2(40)
Einkaufsdatum	date

RechnungsPos	Datentyp
RechnungsNr	number(6)
ArtikelNr	number(6)
Anzahl	number(4)

Foreign Key's

TSBE Bern
Höhere Fachschule

- Foreign Key's oder Fremdschlüssel referenzieren einen Datensatz in einer anderen Tabelle
- Durch das definieren eines FK Constraints stellt die Datenbank sicher, dass keine falschen Werte in die Tabellen geschrieben werden können
- Es wird z.B. geprüft, ob der Kunde mit der Kundennummer 99 existiert bevor eine Rechnung für ihn erstellt werden kann

SQL create table Syntax 1/4

TSBE Bern
Höhere Fachschule

- Einfachste Möglichkeit:

```
create table [tablename] ([Attribut] [Datatype], [Attribut] [Datatype]);
```

- Beispiel:

```
create table mytest (mytest_pk number(5), name varchar2(20));
```

SQL create table Syntax 2/4

TSBE Bern
Höhere Fachschule

- Mit Angabe eines Primary Key's:

```
create table [tablename] ([Attribut] [Datatype],
                          [Attribut] [Datatype],
                           Primary Key ([PK Attribut])
                          );
```

- Beispiel

```
create table mytest2 (mytest2_pk number(5),
                      name       varchar2(20),
                      Primary Key (mytest2_pk)
                          );
```

SQL create table Syntax 3/4

TSBE Bern
Höhere Fachschule

- NOT NULL
  - Mit der Option NOT NULL wird erzwungen, dass ein Attribut auch tatsächlich gefüllt wird
  - Spalten sollten wann immer möglich gefüllt werden. Keine «Excel Sheet Tabellen» machen!

```
create table [tablename] ([Attribut] [Datatype] NOT NULL,
                          [Attribut] [Datatype] NOT NULL
                         );
```

- Beispiel:

```
create table mytest (mytest_pk number(5) NOT NULL,
                     name varchar2(20) NOT NULL
                    );
```

SQL create table Syntax 4/4

TSBE Bern
Höhere Fachschule

- DEFAULT
  - Es ist möglich jedem Attribut einen Defaultwert zu geben
  - Dieser Wert wird bei jedem Insert eingefügt, falls kein anderer Wert explizit angegeben wird

```
create table [tablename] ([Attribut] [Datatype],
                          [Attribut] [Datatype] default 0
                         );
```

- Beispiel

```
create table mytest2 (mytest2_pk    number(5),
                      name          varchar2(20) default 'Mueller',
                      creation_date date         default sysdate
                      );
```

SQL Foreign Key Constraint

TSBE Bern
Höhere Fachschule

- FK Constraints können auch nach dem Erstellen der Tabellen definiert werden

```
alter table [tablename]
  add foreign key ([Attribut])
      references [tablename]([Attribut]);
```

- Beispiel

```
alter table RECHNUNGEN
  add foreign key (kde_nr)
      references kunden(kde_nr);
```

Kleine Ausschweifung

TSBE Bern
Höhere Fachschule

- Datenbank Benutzer
  - Eigene Tabellen sollten nie in DB internen Schemen angelegt werden
  - Um Tabellen erstellen zu können, muss also ein Datenbankbenutzer angelegt werden
- Tablespaces
  - Bei ORACLE werden die Tabellen in «**Tablespaces**» abgelegt
  - Tablespaces sind logische Konstrukte, welche auf physischen Files, den **Datenfiles**, basieren

- → Um unsere Webshop Tabellen anlegen zu können, brauchen wir einen Tablespace sowie einen DB User

Schema’s

TSBE Bern
Höhere Fachschule

- Datenbankobjekte (Tabellen, Indexe, Views, Prozeduren etc.) werden unter einem Datenbankbenutzer angelegt
- Dieser User wird «**Schemaowner**» genannt
- Eine zusammengehörende Gruppe von Datenbankobjekten eines Schemaowners wird «**Schema**» genannt
- Ein Schemaowner besitzt höhere Berechtigungen innerhalb einer Datenbank als ein Zugriffsuser
- Jedem User in einer ORACLE Datenbank wird ein Default Tablespace zugewiesen

Erstellen des Users WEBSHOP 1/2

TSBE Bern
Höhere Fachschule

- Erst muss ein Tablespace erstellt werden

```
create tablespace WEBSHOP_DATA
  datafile '/u02/oradata/XE112/webshop_data01XE112.dbf'
  size 50M
  autoextend on;
```

Erstellen des Users WEBSHOP 2/2

TSBE Bern
Höhere Fachschule

- Anschliessend muss der Schemaowner mit den entsprechenden Berechtigungen angelegt werden

```
create user WEBSHOP
  default tablespace WEBSHOP_DATA
  identified by manager;

grant connect, resource to WEBSHOP;

alter user WEBSHOP quota unlimited on WEBSHOP_DATA;
```

Erstellen des Schema's WEBSHOP 1/6

TSBE Bern
Höhere Fachschule

- Achtung! Vor dem Erstellen entweder erst als User webshop einloggen…

```
SQL> connect webshop/manager
```

- …oder der Tabelle den Schemaowner voranstellen

```
create table webshop.KUNDEN (Kde_Nr     number(6)     NOT NULL,
                             Name       varchar2(50)  NOT NULL,
                             Vorname    varchar2(50)  NOT NULL,
                             Adresse    varchar2(70)  NOT NULL,
                             PLZ        number(5)     NOT NULL,
                             Ort        varchar2(40)  NOT NULL,
                             eMail      varchar2(60)  NOT NULL,
                             Handy      varchar2(13),
                             Telefon    varchar2(13),
                             Primary Key (Kde_Nr)
                            );
```

Erstellen des Schema's WEBSHOP 2/6

TSBE Bern
Höhere Fachschule

- Tabelle Kunden

```
create table KUNDEN (Kde_Nr    number(6)     NOT NULL,
                     Name      varchar2(50)  NOT NULL,
                     Vorname   varchar2(50)  NOT NULL,
                     Adresse   varchar2(70)  NOT NULL,
                     PLZ       number(5)     NOT NULL,
                     Ort       varchar2(40)  NOT NULL,
                     eMail     varchar2(60)  NOT NULL,
                     Handy     varchar2(13),
                     Telefon   varchar2(13),
                     Primary Key (Kde_Nr)
                    );
```

Erstellen des Schema's WEBSHOP 3/6

TSBE Bern
Höhere Fachschule

- Tabelle Artikel

```
create table ARTIKEL (Art_Nr         number(6)                NOT NULL,
                      Bezeichnung    varchar2(50)             NOT NULL,
                      Preis          number(7,2)              NOT NULL,
                      Lagerbestand   number(3)    default 0   NOT NULL,
                      Lagerort       varchar2(50)             NOT NULL,
                      Lieferant      varchar2(40)             NOT NULL,
                      Einkaufsdatum  date         default sysdate,
                      Primary Key (Art_Nr)
                     );
```

Erstellen des Schema's WEBSHOP 4/6

TSBE Bern
Höhere Fachschule

- Tabelle Rechnungen

```
create table RECHNUNGEN (Rg_Nr      number(6)                 NOT NULL,
                         Rg_Datum   date     default sysdate NOT NULL,
                         Kde_Nr     number(6)                NOT NULL,
                         Zahl_frist number(2)   default 30,
                         Zahl_dat   date        default NULL,
                         Betrag     number(8,2) default 0    NOT NULL,
                         Primary Key (Rg_Nr)
                        );
```

Erstellen des Schema's WEBSHOP 5/6

TSBE Bern
Höhere Fachschule

- Tabelle Rechnungspositionen

```
create table RECH_POS (Rg_Nr      number(6) NOT NULL,
                       Art_Nr     number(6) NOT NULL,
                       Anzahl     number(4) NOT NULL,
                       Primary Key (Rg_Nr,Art_Nr)
                      );
```

Erstellen des Schema's WEBSHOP 6/6

TSBE Bern
Höhere Fachschule

- Abfüllen der Tabellen
  - Dies ist nur ein Beispiel.
  - Bitte die Tabellen mit dem Script «07_inserts.sql» laden

```
insert into kunden (kde_nr,
                    name,
                    vorname,
                    adresse,
                    plz,
                    ort,
                    email,
                    handy)
            values (1,
                    'Mueller',
                    'Peter',
                    'Sustenweg 22',
                    3013,
                    'Bern',
                    'peter.mueller@gmail.com',
                    '079/123 45 67'
                   );
```

Abfragen der Tabellen 1/3

TSBE Bern
Höhere Fachschule

- Mit folgendem select Statement können Informationen zur Rechnung Nr 1 ausgegeben werden

```
select k.name,
       k.vorname,
       r.rg_nr,
       r.rg_datum,
       a.bezeichnung,
       rp.anzahl,
       a.preis,
       a.preis * rp.anzahl totalpreis
from   kunden k,
       rechnungen r,
       artikel a,
       rech_pos rp
where  k.kde_nr = r.kde_nr
  and  r.rg_nr = rp.rg_nr
  and  a.art_nr = rp.art_nr
  and  r.rg_nr = 1
order by a.bezeichnung;
```

- Man beachte die Joins über die 4 Tabellen

Abfragen der Tabellen 2/3

TSBE Bern
Höhere Fachschule

- Mit folgendem select Statement wird der Totalbetrag der Rechnung Nr 1 ausgegeben

```
select sum(a.preis*rp.anzahl) rechnungsbetrag
from   artikel a,
       rech_pos rp
where  a.art_nr = rp.art_nr
  and  rp.rg_nr = 1;
```

Abfragen der Tabellen 3/3

TSBE Bern
Höhere Fachschule

- Dieser Statement kombiniert die beiden vorigen Statements
- Interessant hier ist der Subselect. Ein Select Statement innerhalb eines Select Statements

```
select k.name,
       k.vorname,
       r.rg_nr,
       r.rg_datum,
       a.bezeichnung,
       rp.anzahl,
       a.preis,
       a.preis * rp.anzahl totalpreis,
       ss.rechnungsbetrag
from   kunden k,
       rechnungen r,
       artikel a,
       rech_pos rp,
       (select sum(aa.preis*rrp.anzahl) rechnungsbetrag
        from   artikel aa,
               rech_pos rrp
        where  aa.art_nr = rrp.art_nr
          and  rrp.rg_nr = &rg_nr
        ) ss
where   k.kde_nr = r.kde_nr
  and   r.rg_nr = rp.rg_nr
  and   a.art_nr = rp.art_nr
  and   r.rg_nr=&rg_nr
order by a.bezeichnung;
```

Fragen?

TSBE Bern
Höhere Fachschule