Autohändler

CREATE TABLE presentation (id int PRIMARY KEY AUTO_INCREMENT, name VARCHAR(50), progress DECIMAL(5,2) DEFAULT 0);

INSERT INTO presentation (name) VALUES ('Marcus Bauer, Alexander Hahn');

Inhalt

- Ziele unserer Datenbank
- Anforderungsanalyse
- ERM
- Create-Statements für
  - Auto
  - Bestellung
  - Posten

Ziele unserer Datenbank

- Ein Kunde soll beliebig viele Autos kaufen, welche sich durch verschiedene Attribute (Baujahr, PS, Modell, Hersteller, Farbe, Typ) kennzeichnen
- Ebendieses System soll für Autohändler realisiert werden, die Kunden, deren Bestellungen und den aktuellen Bestand an Autos verwalten können.

Anforderungsanalyse

- Grundsätzliches Element der Autohändler-Datenbank ist das Auto
- Für zwei grundsätzlich gleiche Autos mit unterschiedlichen Attributen existieren zwei verschiedene Einträge in der Datenbank

Objekt:	Auto					
Attribut	Typ	Länge	Bereich	Wiederholung	Definiertheit	Identifizierend
id	int	5	1…99999	0	100%	ja
Hersteller_id	int	5	1…99999	0	100%	nein
Typ_id	int	5	1…99999	0	100%	nein
Bezeichnung	String			0	100%	nein
Baujahr	int	4	1900+	0	100%	nein
Distanz	Decimal	10,1	0+	0	100%	nein
PS	int	3	1…999	0	100%	nein
Verbrauch	Decimal	3,1	1…99	0	100%	nein
Lagerbestand	int	3	0+	0	100%	nein
Preis	Decimal	10,2	0+	0	100%	nein

Anforderungsanalyse II

- Auto hat verschiedene Attribute

Objekt:	Hersteller					
Attribut	Typ	Länge	Bereich	Wiederholung	Definiertheit	Identifizierend
id	int	5	1...99999	0	100%	ja
Name	String	1...50		0	100%	nein

Objekt:	Typ					
Attribut	Typ	Länge	Bereich	Wiederholung	Definiertheit	Identifizierend
id	int	5	1...99999	0	100%	ja
Name	String	1...50		0	100%	nein

Objekt:	Farbe					
Attribut	Typ	Länge	Bereich	Wiederholung	Definiertheit	Identifizierend
id	int	5	1...99999	0	100%	ja
Name	String	1...50		0	100%	nein

Objekt:	Autofarbe					
Attribut	Typ	Länge	Bereich	Wiederholung	Definiertheit	Identifizierend
Auto_id	int	5	1...99999	1	100%	ja
Farb_id	int	5	1...99999	0	100%	ja

Anforderungsanalyse III

- Kunde als Empfänger eines verkauften Autos
- Name und Adresse werden gespeichert

Objekt:	Kunde					
Attribut	Typ	Länge	Bereich	Wiederholung	Definiertheit	Identifizierend
id	int	5	1...99999	0	100%	ja
Name	String	1...50		0	100%	nein
Adresse	String	1...50		1	100%	nein
Telefon	String	1...50		1	100%	nein
E-Mail	String	1...50		1	80%	nein

Anforderungsanalyse IV

- Autokauf erfolgt über eine Bestellung
- Zustand ändert sich, sobald das Auto geliefert oder bezahlt wurde

Objekt:	Bestellung					
Attribut	Typ	Länge	Bereich	Wiederholung	Definiertheit	Identifizierend
id	int	5	1…99999	0	100%	ja
Kunde_id	int	5	1…99999	0	100%	nein
Gesamtpreis	decimal	10,2	0+	0	100%	nein
Zahlungsart_id	int	5	1…99999	0	100%	nein
ausgeliefert	int	1	0…1	0	100%	nein
bezahlt	int	1	0…1	0	100%	nein
Bestelldatum	date		1.1.2012 .. *	0	100%	nein

Anforderungsanalyse V

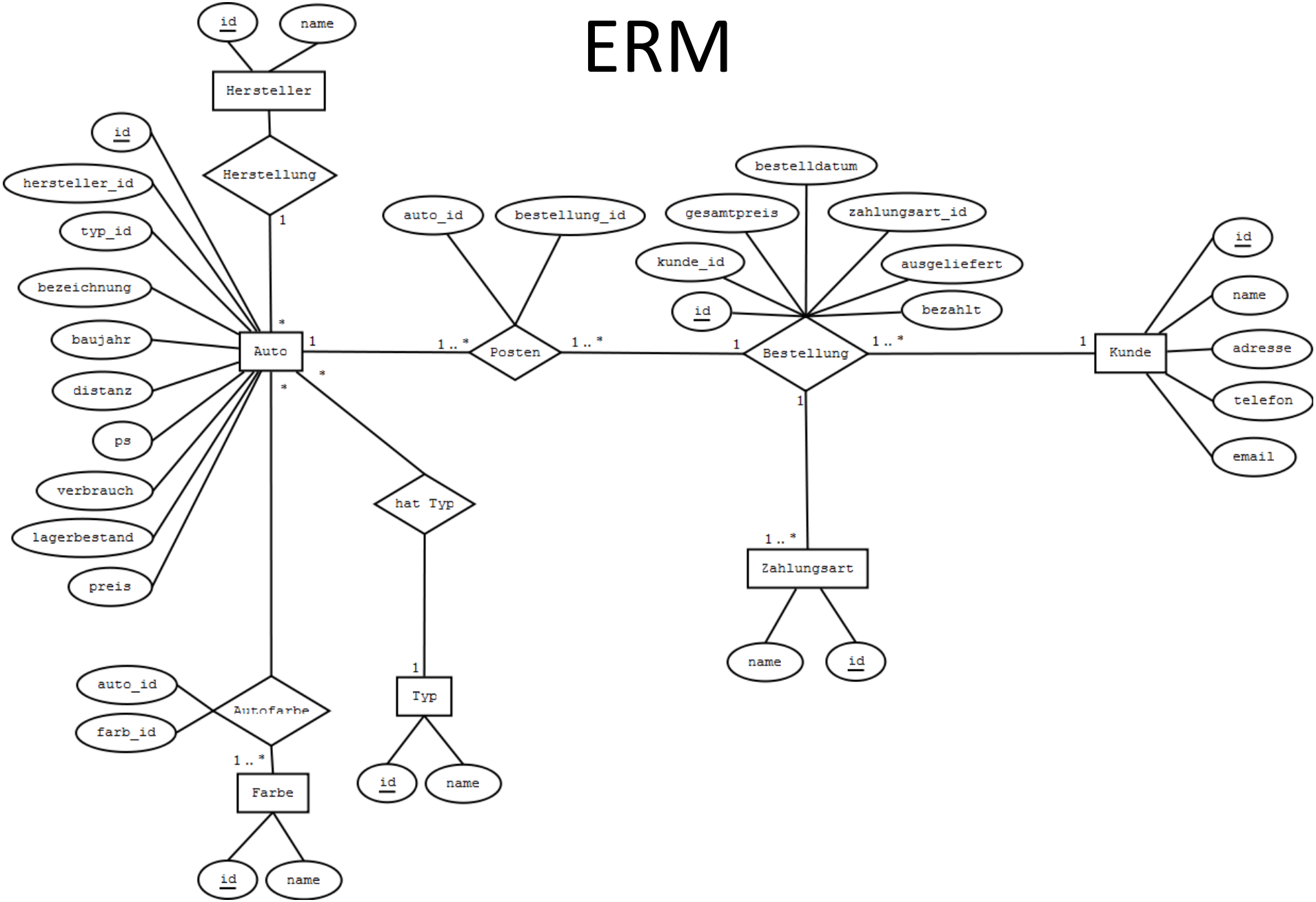
- Eine Bestellung kann mehrere Posten

Objekt:	Posten					
Attribut	Typ	Länge	Bereich	Wiederholung	Definiertheit	Identifizierend
Auto_id	int	5	1...99999	1	100%	ja
Bestellung_id	int	5	1...99999	0	100%	ja

- Die Zahlungsart ist ebenfalls als separate Tabelle existent

Objekt:	Zahlungsart					
Attribut	Typ	Länge	Bereich	Wiederholung	Definiertheit	Identifizierend
id	int	5	1...99999	0	100%	ja
Name	String	1...50		0	100%	nein

ERM


id
name
Hersteller
Herstellung
1
*
id
hersteller_id
typ_id
bezeichnung
baujahr
distanz
ps
verbrauch
lagerbestand
preis
Auto
1
*
*
auto_id
bestellung_id
1 .. *
Posten
1 .. *
bestelldatum
gesamtpreis
zahlungsart_id
kunde_id
ausgeliefert
id
bezahlt
1
Bestellung
1 .. *
1
Kunde
id
name
adresse
telefon
email
1
1 .. *
Zahlungsart
name
id
hat Typ
1
Typ
id
name
auto_id
farb_id
Autofarbe
1 .. *
Farbe
id
name

Create-Statement für Auto

```
CREATE TABLE auto (
  id int(11) NOT NULL AUTO_INCREMENT,
  bezeichnung varchar(50) DEFAULT NULL,
  hersteller_id int(11) NOT NULL,
  baujahr int(4) NOT NULL DEFAULT 2012 CHECK(baujahr >= 1950 AND baujahr <=
YEAR(CURRENT_DATE)),
  distanz decimal(10,1) NOT NULL DEFAULT 0 CHECK(distanz >= 0),
  ps int(3) NOT NULL DEFAULT 100 CHECK(ps > 0),
  verbrauch decimal(3,1) NOT NULL DEFAULT 6.5 CHECK(verbrauch >= 3),
  lagerbestand int(11) NOT NULL DEFAULT 0 CHECK(lagerbestand >= 0),
  preis decimal(10,2) NOT NULL CHECK(preis > 0),
  typ_id int(11) NOT NULL,
  PRIMARY KEY (id),
  KEY hersteller (hersteller_id),
  KEY farbe (farb_id),
  CONSTRAINT farbe FOREIGN KEY (farb_id) REFERENCES farbe (id) ON DELETE CASCADE
ON UPDATE CASCADE,
  CONSTRAINT hersteller FOREIGN KEY (hersteller_id) REFERENCES hersteller (id) ON
DELETE CASCADE ON UPDATE CASCADE
);
```

Create-Statement für Bestellungen

```
CREATE TABLE bestellung (
  id int(11) NOT NULL AUTO_INCREMENT,
  kunde_id int(11) DEFAULT NULL,
  gesamtpreis decimal(10,2) NOT NULL CHECK(gesamtpreis > 0),
  bestelldatum date NOT NULL CHECK (bestelldatum > '2012-01-01'),
  zahlungsart_id int(11) NOT NULL,
  ausgeliefert int(1) NOT NULL DEFAULT '0' CHECK (ausgeliefert IN (0,1),
  bezahlt int(1) NOT NULL DEFAULT '0' CHECK (bezahlt IN (0,1),
  PRIMARY KEY (id),
  KEY kunde (kunde_id),
  CONSTRAINT kunde FOREIGN KEY (kunde_id) REFERENCES kunde (id)
ON DELETE SET NULL ON UPDATE NO ACTION
);
```

Create-Statement für Posten

```
CREATE TABLE posten (
  auto_id int(11) NOT NULL,
  bestellung_id int(11) NOT NULL,
  PRIMARY KEY (auto_id,bestellung_id),
  KEY auto (auto_id),
  KEY bestellung (bestellung_id),
  CONSTRAINT bestellung FOREIGN KEY (bestellung_id) REFERENCES bestellung (id) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT auto FOREIGN KEY (auto_id) REFERENCES auto (id) ON DELETE CASCADE ON UPDATE CASCADE
);
```

Trigger für Posten

- Nur zur Ergänzung
- Falls man ein Auto bestellt, wird die Lagermenge des entsprechenden Autos reduziert.

```
CREATE TRIGGER posten_update AFTER INSERT
ON posten
FOR EACH ROW UPDATE auto SET lagermenge =
lagermenge - 1 WHERE id = auto_id;
```

Ende

```
UPDATE presentation SET progress = 100.00 WHERE name = 'Marcus Bauer, Alexander Hahn';
```