Datenbanken: Datenintegrität

Definition "Datenkonsistenz"

"in der Datenbankorganisation (...) die **Korrektheit der gespeicherten Daten** im Sinn einer **widerspruchsfreien und vollständigen Abbildung** der relevanten Aspekte des erfassten Realitätsausschnitts."

http://wirtschaftslexikon.gabler.de/Archiv/57132/datenintegritaet-v6.html
Hervorhebung hinzugefügt

→ *logisch korrekter Zustand der Daten*

Inkonsistente / konsistente Daten

kunden

idKunde	name	postleitzahl	ortName	arbeitgeber
1	Schmitt	10000	Musterhausen	Bäckerei Zimmermann GmbH
2	Müller	10000	Musterausen	Schlüter & Co KG
	Müller	79132	Coburg	Bäckerei Zimmermann

Welche Fehler erkennen Sie?

(abgesehen von der Verletzung einer Normalform)

Inkonsistente / konsistente Daten

kunden

idKunde	name	postleitzahl	ortName	arbeitgeber
1	Schmitt	10000	Musterhausen	Bäckerei Zimmermann GmbH
2	Müller	10000	Musterausen	Schlüter & Co KG
	Müller	79132	Coburg	Bäckerei Zimmermann

Inkonsistente / konsistente Daten

kunden

idKunde	name	postleitzahl	arbeitgeberFK
1	Schmitt	10000	1
2	Müller	10000	14
3	Maier	79312	7

(FK = Foreign Key, Fremdschlüssel)

orte

postleitzahl	name
10000	Musterhausen
79098	Freiburg

Welche Fehler erkennen Sie?

Inkonsistente / konsistente Daten

kunden

idKunde	name	postleitzahl	arbeitgeberFK
1	Schmitt	10000	1
2	Müller	10000	14
3	Maier	79312	7

orte

postleitzahl	name
10000	Musterhausen
79098	Freiburg

arbeitgeber

idArbeitgeber	name
1	Bäckerei Zimmermann
7	Hug GmbH
14	ForSi

Integritätsregeln

Datenkonsistenz/Datenintegrität wird gewährleistet durch Integritätsbedingungen:

Domänenintegrität/Bereichsintegrität:
Attribute sind nur gültig, wenn sie einen bestimmten Wertebereich haben

id	name	geburtsdatum
1	Smith	hihihi

Entitätsintegrität
Jeder Datensatz ist eindeutig definiert (z.B. durch PRIMARY KEY).

id	name	geburtsdatum
1	Smith	2012-12-12
1	Sponz	2012-12-13

Referentielle Integrität
Beziehungen zwischen Tabellen müssen synchronisiert bleiben. (s.u.)

Benutzerdefinierte Integrität
Sonstige vom Benutzer festgelegte Regeln
(z.B.: Datum darf nicht vor 01.01.2000 liegen)

Integritätsregeln

Datenkonsistenz/Datenintegrität wird gewährleistet durch Integritätsbedingungen:

Domänenintegrität/Bereichsintegrität:

Entitätsintegrität

Referentielle Integrität

Benutzerdefinierte Integrität

Sind diese Bedingungen erfüllt, ist die Datenbank konsistent.

Konsistente Transformation: COMMIT / ROLLBACK

Jede Transaktion muss eine Datenbank von einem konsistenten in einen anderen konsistenten Zustand überführen. Während der Verarbeitung der Anfrage kann die Konsistenz der Datenbank jedoch kurzfristig verletzt werden.

Nach jeder durch eine Transaktion gegebene Reihe von Veränderungen der Daten (Einfügen, Löschen oder Ändern) wird die Datenbank auf die Integritätsbedingungen geprüft. Falls diese nicht erfüllt sein sollten, muss die gesamte Transaktion so zurück abgewickelt werden, dass der vorige (konsistente) Zustand wiederhergestellt wird („Rollback“).

http://de.wikipedia.org/wiki/Konsistenz_(Datenspeicherung)#Konsistente_Transformationen

mitarbeiter

mitarbeiter	gehalt
1	1000
2	1000
3	1000
4	1000

```
UPDATE mitarbeiter SET gehalt = 5000;
```

DBMS crasht WÄHREND der Ausführung, Ergebnis:

mitarbeiter	gehalt
1	5000
2	5000
3	5000
4	1000

(Unvollständige) Transaktion kann mit ROLLBACK wieder rückgängig gemacht werden.

Referentielle Integrität

Primärschlüssel-Fremdschlüssel-Beziehungen müssen intakt sein:

kunden

idKunde	name	postleitzahl	arbeitgeberFK
1	Schmitt	10000	1
2	Müller	10000	14
3	Maier	79312	7

Keine passende Tabelle
→ kein PK zu den FKs

Kein passender Datensatz
→ kein PK zum FK

orte

postleitzahl	name
10000	Musterhausen
79098	Freiburg

möglicherweise nett fürs Abi ☺

Definition
"Die referentielle Integrität (auch Beziehungsintegrität) besagt, dass Attributwerte eines Fremdschlüssels auch als Attributwert des Primärschlüssels vorhanden sein müssen."

http://de.wikipedia.org/wiki/Referentielle_Integrität

Referentielle Integrität

Primärschlüssel-Fremdschlüssel-Beziehungen müssen intakt sein:

kunden

idKunde	name	postleitzahl	arbeitgeberFK
1	Schmitt	10000	1
2	Müller	10000	14
3	Maier	79312	7

Keine passende Tabelle
→ kein PK zu den FKs

Kein passender Datensatz
→ kein PK zum FK

orte

postleitzahl	name
10000	Musterhausen
79098	Freiburg

Grundregel

Fremdschlüssel müssen IMMER auf existierende Datensätze verweisen!

Referentielle Integrität

Grundregel

Fremdschlüssel müssen IMMER auf existierende Datensätze verweisen!

kunden

idKunde	name	postleitzahl
1	Schmitt	10000
2	Müller	10000
3	Maier	79312

muss beachtet werden bei ...

1. Löschen von Datensätzen oder Tabellen

orte

postleitzahl	name
~~10000~~	~~Musterhausen~~
79312	Emmendingen

Löschen dieses Datensatzes führt zu Inkonsistenzen (da zum FK "kunden.postleitzahl" kein Datensatz existiert)

Referentielle Integrität

Grundregel

Fremdschlüssel müssen IMMER auf existierende Datensätze verweisen!

kunden

idKunde	name	postleitzahl
1	Schmitt	10000
2	Müller	10000
3	Maier	79312
4	Huber	80985

muss beachtet werden bei ...

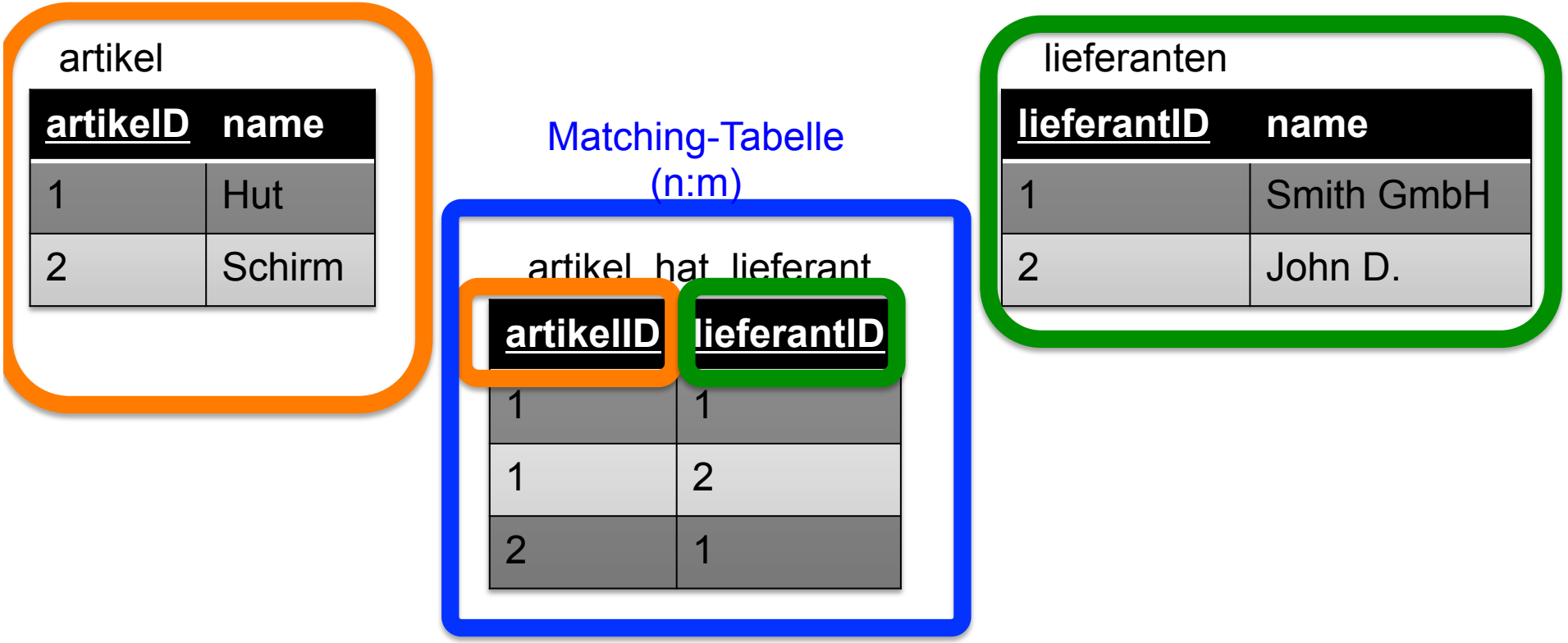
2. Einfügen von Datensätzen

orte

postleitzahl	name
10000	Musterhausen
79312	Emmendingen

Einfügen dieses Datensatzes führt zu Inkonsistenzen (da zum FK "kunden.postleitzahl" kein Datensatz existiert)

Aufgaben


artikel
artikelID
name
1
Hut
2
Schirm
Matching-Tabelle
(n:m)
artikel_hat_lieferant
artikelID
lieferantID
1
1
1
2
2
1
lieferanten
lieferantID
name
1
Smith GmbH
2
John D.

Aufgaben

artikel

artikelID	name
1	Hut
2	Schirm
3	Schuh

(a)

artikel_hat_lieferant

artikelID	lieferantID
1	1
1	2
2	1
2	3

(c)

(b)

lieferanten

lieferantID	name
1	Smith GmbH
2	John D.

(d)

Welche Operationen sind zulässig?

a) Einfügen Datensatz
b) Einfügen Datensatz
c) Löschen Datensatz
d) Löschen Datensatz

Änderungsweitergabe, Löschweitergabe; MySQL: FOREIGN-KEY-CONSTRAINTS

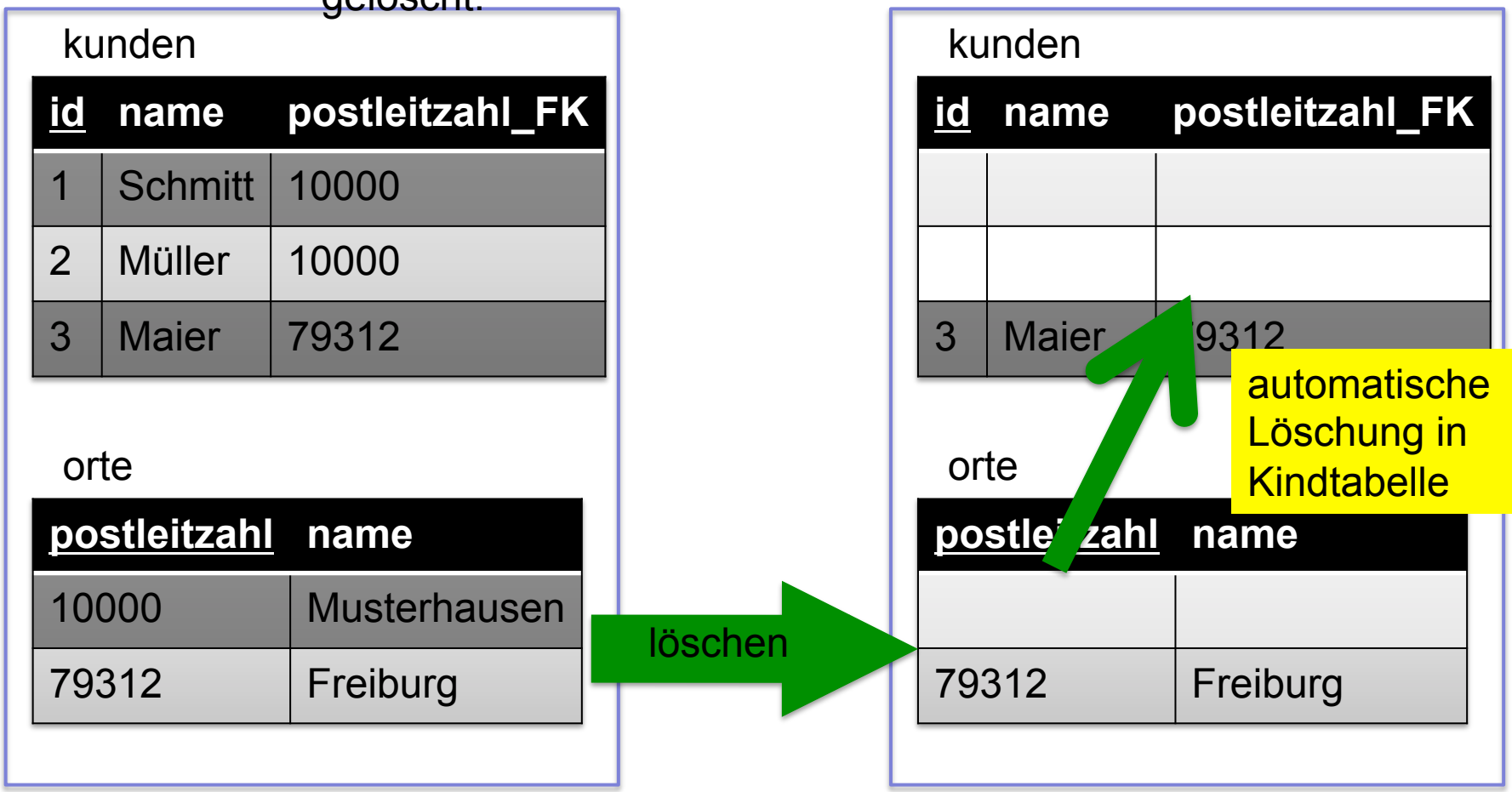
DBMS kann helfen, Datenintegrität zu wahren, z.B. über Einschränkungen (CONSTRAINTS):

- Änderungsweitergabe
- Löschweitergabe

Achtung:
InnoDB erforderlich!

Löschweitergabe

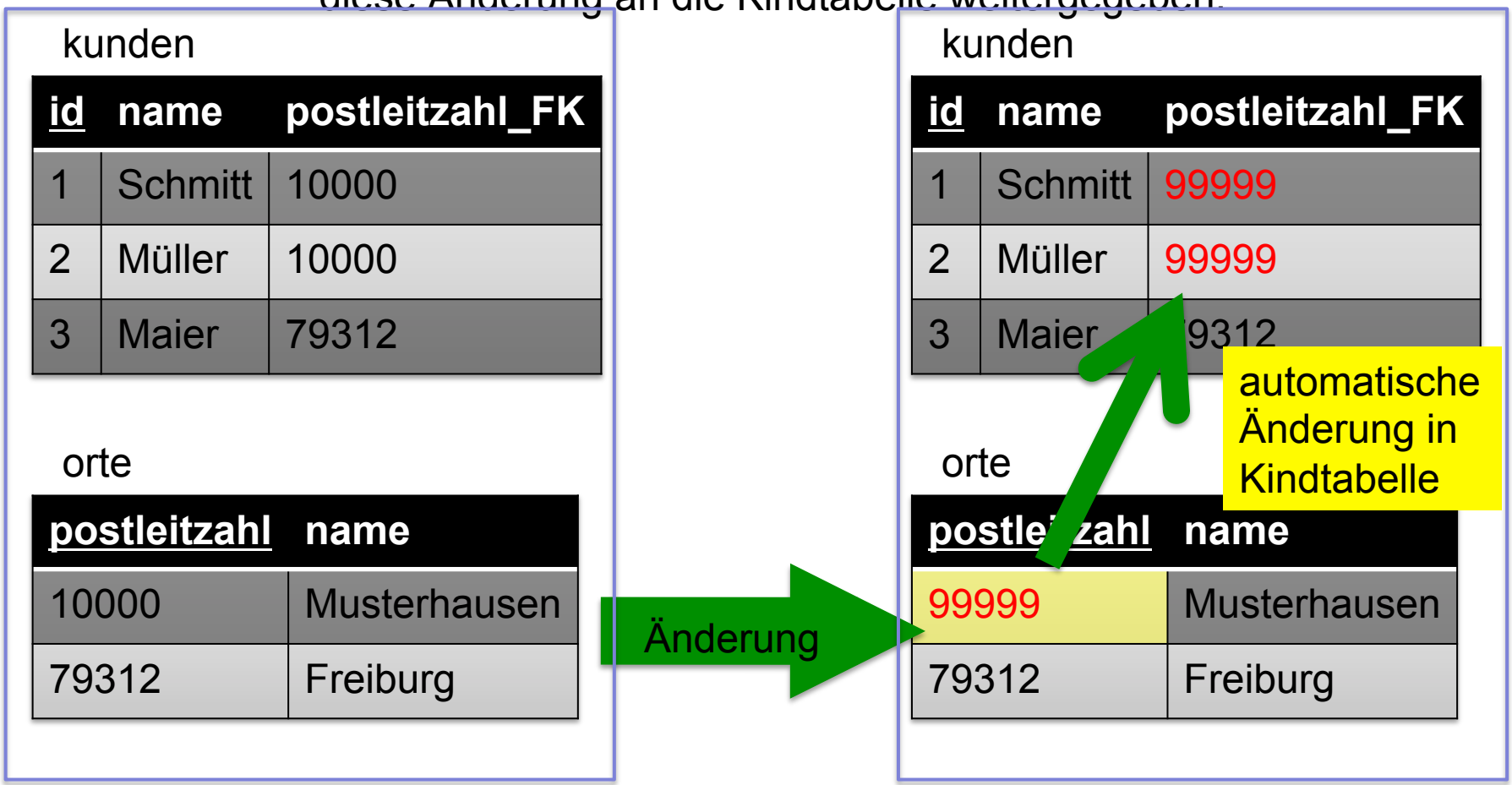
Löschen wir einen Ort, dann werden alle Kunden in diesem Ort gelöscht.



(Das ist natürlich nicht sinnvoll; besseres Beispiel: Wird ein Lieferant gelöscht, dann könnte man auch die von ihm gelieferten Artikel löschen.

Änderungsweitergabe

Ändert sich die PLZ von Musterhausen (Elterntabelle "orte"), wird diese Änderung an die Kindtabelle weitergegeben.



Änderungsweitergabe

Ändert sich die PLZ von Musterhausen (Elterntabelle "orte"), wird diese Änderung an die Kindtabelle weitergegeben.

kunden

id	name	postleitzahl_FK
1	Schmitt	10000
2	Müller	10000
3	Maier	79312

kunden

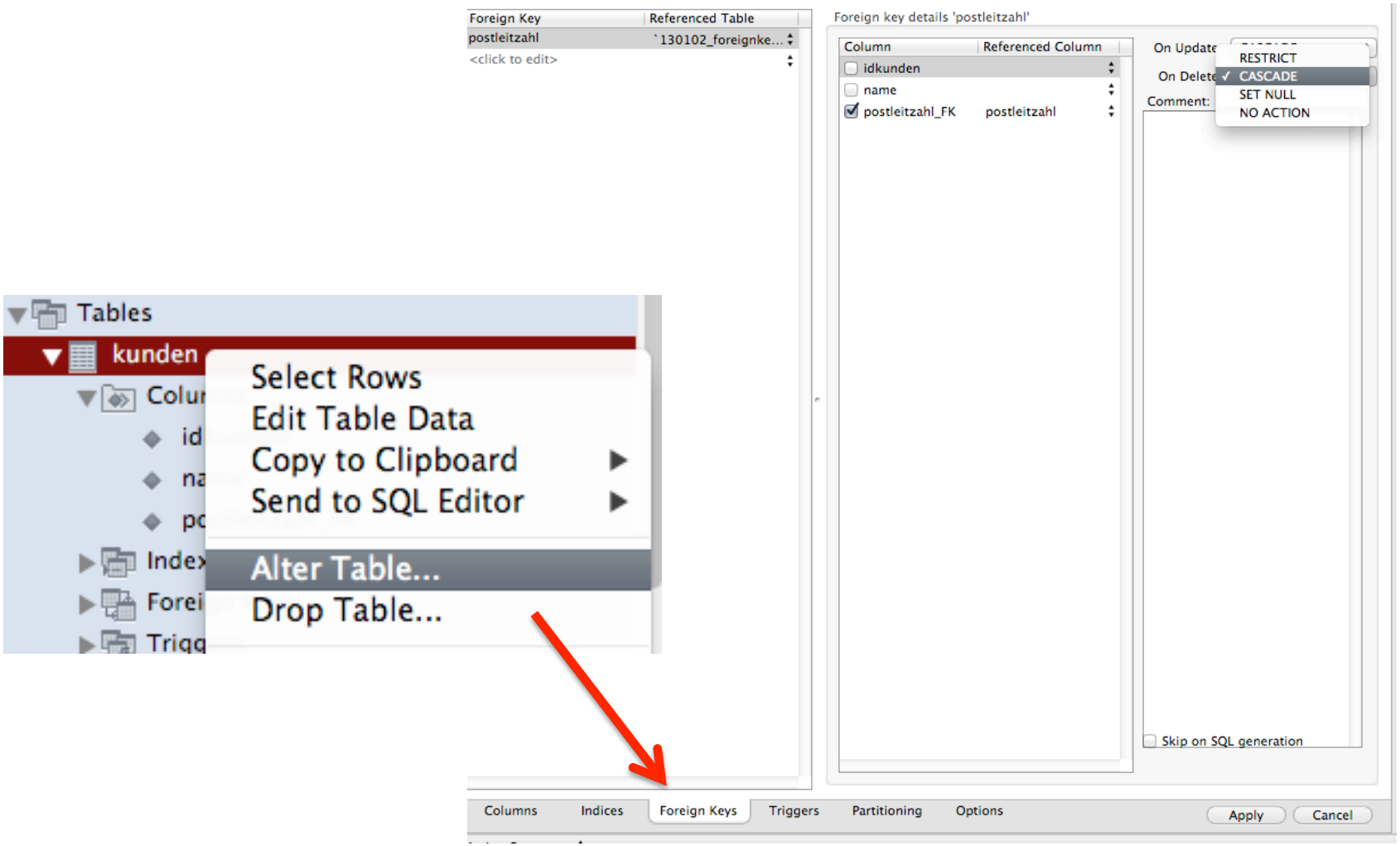
id	name	postleitzahl_FK
1	Schmitt	99999
2	Müller	99999
3	Maier	79312

automatische Änderung in Kindtabelle

```
CREATE  TABLE kunden (
  id INT, name VARCHAR(45),postleitzahl_FK VARCHAR(5),
  PRIMARY KEY (idkunden), -- ....
    CONSTRAINT FK_postleitzahl
        FOREIGN KEY (postleitzahl_FK)
            REFERENCES orte (postleitzahl)
        ON DELETE NO ACTION
        ON UPDATE CASCADE)
ENGINE = InnoDB;
```

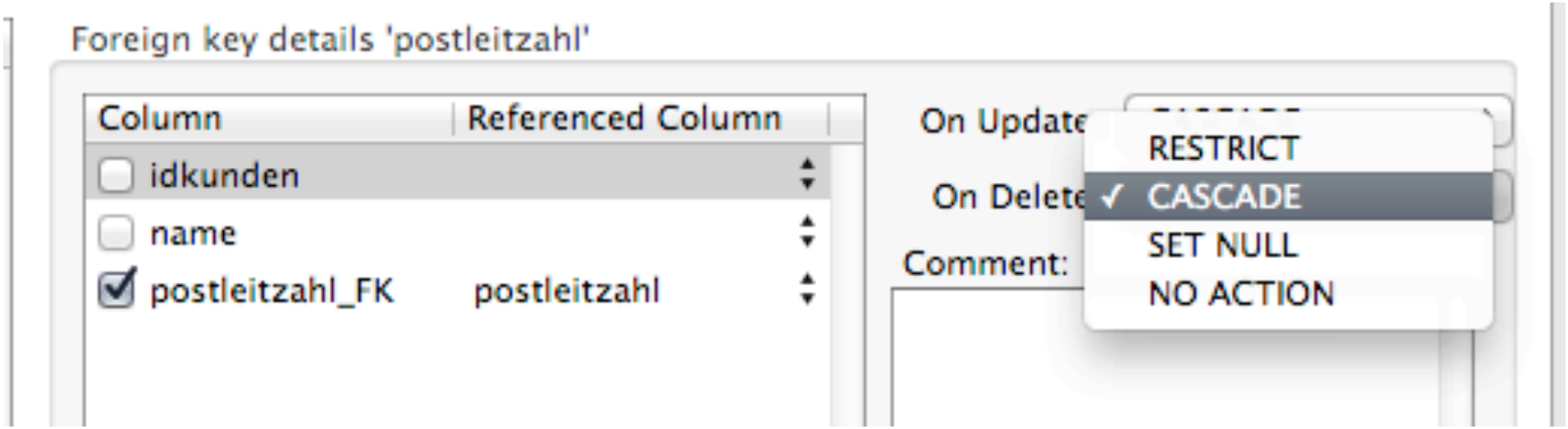
Änderungsweitergabe

Ändert sich die PLZ von Musterhausen (Elterntabelle "orte"), wird diese Änderung an die Kindtabelle weitergegeben.



Änderungsweitergabe

Ändert sich die PLZ von Musterhausen (Elterntabelle "orte"), wird diese Änderung an die Kindtabelle weitergegeben.



FOREIGN KEY-CONSTRAINTS

MySQL/InnoDB

```
ON UPDATE ...
ON DELETE ...
```

CASCADE	Änderungen/Löschung auch in Kindtabelle vornehmen
SET NULL	Fremdschlüssel in Kindtabelle auf NULL setzen
NO ACTION	keine weitere Aktion
RESTRICT	Aktion verweigern

FOREIGN KEY-CONSTRAINTS

MySQL/InnoDB

```
ON UPDATE ...
ON DELETE ...
```

CASCADE	Änderungen/Löschung auch in Kindtabelle vornehmen
SET NULL	Fremdschlüssel in Kindtabelle auf NULL setzen
NO ACTION	keine weitere Aktion
RESTRICT	

```
CREATE  TABLE kunden (
  id INT, name VARCHAR(45),postleitzahl_FK VARCHAR(5),
  PRIMARY KEY (idkunden), -- ....
    CONSTRAINT FK_postleitzahl
        FOREIGN KEY (postleitzahl_FK)
            REFERENCES orte (postleitzahl)
        ON DELETE NO ACTION
        ON UPDATE CASCADE)
ENGINE = InnoDB;
```

Aufgaben

Benutzen Sie datenmodellierung05_datenintegritaet_kundendump.sql oder kopieren Sie den Code rechts (2px-Schrift).

-- CODE zum Kopieren:

```
DROP SCHEMA IF EXISTS `130102_foreignkey-tests` ;
CREATE SCHEMA IF NOT EXISTS `130102_foreignkey-tests` DEFAULT CHARACTER SET latin1 ;

USE `130102_foreignkey-tests` ;

-- -----------------------------------------------------
-- Table `130102_foreignkey-tests`.`kunden`
-- -----------------------------------------------------


-- -----------------------------------------------------
-- Table `130102_foreignkey-tests`.`orte`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `130102_foreignkey-tests`.`orte` ;

CREATE  TABLE IF NOT EXISTS `130102_foreignkey-tests`.`orte` (
  `postleitzahl` VARCHAR(5) NOT NULL ,
  `name` VARCHAR(45) NULL DEFAULT NULL ,
  PRIMARY KEY (`postleitzahl`) )
ENGINE = InnoDB
DEFAULT CHARACTER SET = latin1;

LOCK TABLES `orte` WRITE;
INSERT INTO `orte` VALUES ('10000','Musterhausen'),('79092','Freiburg'),('79102','Freiburg'),
('79312','Emmendingen');
UNLOCK TABLES;


DROP TABLE IF EXISTS `130102_foreignkey-tests`.`kunden` ;

CREATE  TABLE IF NOT EXISTS `130102_foreignkey-tests`.`kunden` (
  `idkunden` INT(11) NOT NULL AUTO_INCREMENT ,
  `name` VARCHAR(45) NULL DEFAULT NULL ,
  `postleitzahl_FK` VARCHAR(5) NOT NULL ,
  PRIMARY KEY (`idkunden`) ,
  INDEX `postleitzahl_FK_idx` (`postleitzahl_FK` ASC) ,
  CONSTRAINT `postleitzahl_FK`
    FOREIGN KEY (`postleitzahl_FK` )
    REFERENCES `130102_foreignkey-tests`.`orte` (`postleitzahl` )
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
AUTO_INCREMENT = 7
DEFAULT CHARACTER SET = latin1;

--
-- Dumping data for table `kunden`
--

LOCK TABLES `kunden` WRITE;
INSERT INTO `kunden` VALUES (1,'Huber','10000'),(2,'Schmitt','10000'),(3,'Müller','79312'),(4,'Maier','79312'),
(5,'Metz','79092'),(6,'Schmied','79102');
UNLOCK TABLES;
```

1. Löschen Sie alle Orte (**`DELETE FROM orte;`**)
2. Wenden Sie **`RESTRICT`** auf **`ON UPDATE`** und **`ON DELETE`** an.
3. Löschen Sie via SQL-Befehl den Ort Emmendingen (**`DELETE FROM orte WHERE`** ...).
4. Ändern Sie die Postleitzahl von Musterhausen (**`UPDATE orte SET postleitzahl = 99999 WHERE`** ...).
5. Sie haben festgestellt, dass sich Änderungen nicht durchführen lassen. Ändern Sie die FOREIGN-KEY-CONSTRAINTS und probieren Sie sie bspw. mit Nr. 3 und 4 oben aus.

Möglicherweise funktionieren einige Aufgaben nicht – warum nicht?