INFORMATIONSSYSTEME

PETER KOVAR

Contents

1. Basic Script

```
CREATE DATABASE pizzabase;
USE pizzabase;

CREATE TABLE kunden(
id INT(3) PRIMARY KEY,
vorname VARCHAR(15) NOT NULL,
nachname VARCHAR(15) NOT NULL,
email VARCHAR(25),
num_mobil VARCHAR(15) NOT NULL);

INSERT INTO kunden VALUES
(1, 'Dennis', 'Schuh', 'dennis@htl.at','0120105');
INSERT INTO kunden VALUES
(2, 'Wanda', 'Schuh', 'wanda@htl.at','0120105');
INSERT INTO kunden VALUES
(3, 'Sue', 'Peh', 'sue@htl.at','0158801');

CREATE TABLE pizzas(
id INT(3) PRIMARY KEY,
name VARCHAR(20) NOT NULL,
preis DECIMAL(4,2) NOT NULL);

INSERT INTO pizzas VALUES
(1, 'Hawaii', 9.80),
(2, 'Salami', 11.50),
(3, 'Tonno', 7.80),
(4, 'Diabolo', 12.90);
```

```
CREATE TABLE bestellungen(
best_nr INT(5) PRIMARY KEY AUTO_INCREMENT,
datum date NOT NULL,
kunden_id INT(3) NOT NULL,
pizzas_id INT(3) NOT NULL,
FOREIGN KEY(kunden_id) REFERENCES kunden_id(id)
ON UPDATE CASCADE
ON DELETE NO ACTION,
FOREIGN KEY(pizzas_id) REFERENCES pizzas(id)
ON UPDATE CASCADE
ON DELETE NO ACTION
);

INSERT INTO bestellungen (datum, kunden_id, pizzas_id)
VALUES (CURRENT_DATE(), 1, 1);
INSERT INTO bestellungen (datum, kunden_id, pizzas_id)
VALUES (CURRENT_DATE(), 1, 2);
INSERT INTO bestellungen (datum, kunden_id, pizzas_id)
VALUES (CURRENT_DATE(), 1, 4);
INSERT INTO bestellungen (datum, kunden_id, pizzas_id)
VALUES (CURRENT_DATE(), 2, 2);
INSERT INTO bestellungen (datum, kunden_id, pizzas_id)
VALUES (CURRENT_DATE(), 2, 1);
INSERT INTO bestellungen (datum, kunden_id, pizzas_id)
VALUES (CURRENT_DATE(), 2, 3);
INSERT INTO bestellungen (datum, kunden_id, pizzas_id)
VALUES (CURRENT_DATE(), 3, 1);
INSERT INTO bestellungen (datum, kunden_id, pizzas_id)
VALUES (CURRENT_DATE(), 3, 2);
INSERT INTO bestellungen (datum, kunden_id, pizzas_id)
VALUES (CURRENT_DATE(), 3, 3);
```

2. Joins

Nachdem das Basic-Skript in die Datenbank geladen wurde können wir mit den Joins beginnen.

2.1. **Wer hatte welche Pizza.**

```
select kunden.vorname, kunden.nachname, pizzas.name
from bestellungen
join kunden on bestellungen.kunden_id = kunden.id
join pizzas on bestellungen.pizzas_id = pizzas.id;
```

2.2. **Wer hatte welche Pizza mit Preis.**

```
select kunden.vorname, kunden.nachname, pizzas.name, pizzas.preis
from bestellungen
join kunden on bestellungen.kunden_id = kunden.id
join pizzas on bestellungen.pizzas_id = pizzas.id;
```

2.3. **Wieviel ist der Umsatz pro Kunde.**

```
select kunden.vorname, kunden.nachname, sum(pizzas.preis)
from bestellungen
join kunden on bestellungen.kunden_id = kunden.id
join pizzas on bestellungen.pizzas_id = pizzas.id
group by kunden.id;
```

3. Referentielle Integritt

Die ersten einfachen Joins haben ja bereits recht gut funktioniert, jetzt ist es an der Zeit die Integritt der Daten zu Testen. Wir haben festgelegt welche Auswirkungen die Befehle `UPDATE` und `DELETE` auf die verknpften Bestelldaten haben sollen.