DB-Dokumentation – Bestelltool

1. Projektüberblick

Das Projekt „Ordermanagement by TiHo“ ist ein webbasiertes Bestellsystem, entwickelt mit Flask (Python) und MariaDB 10.11. Es bietet Kunden die Möglichkeit, sich zu registrieren, anzumelden, einen Artikelkatalog einzusehen, Bestellungen aufzugeben und ihre Bestellhistorie einzusehen. Admins können Artikel und Bestellungen verwalten, Artikel hinzufügen oder ändern und Bestellungen als Excel-Dateien exportieren. Das System wird auf einem Raspberry Pi gehostet und ist öffentlich unter https://webserver-pironman.duckdns.org/ erreichbar.

2. Fachliche und technische Anforderungen

Fachlich:

- Kunden können Artikel sehen, bestellen und frühere Bestellungen einsehen.
- Kunden können Artikel nachbestellen.
- Admins können Artikel und Bestellungen verwalten.

Technisch:

- Einsatz einer relationalen Datenbank (MariaDB 10.11).
- Umsetzung mit Flask (Python) als Backend.
- Verwendung von HTTPS für sicheren Datentransport.
- Passwort-Hashing für sichere Speicherung von Zugangsdaten.

3. Datenbankentwurf & Schlüssel-Beziehungen

Primärschlüssel (PK): Eine eindeutige ID für jede Zeile, vergleichbar mit einer Ausweisnummer. **Fremdschlüssel (FK):** Ein Verweis auf einen Eintrag in einer anderen Tabelle, der die Verbindung zwischen den Tabellen herstellt.

Beziehungen:

- 1:n – Ein Kunde kann viele Bestellungen haben, aber jede Bestellung gehört zu genau einem Kunden.
- m:n – Eine Bestellung kann viele Artikel enthalten, und ein Artikel kann in vielen Bestellungen vorkommen. Diese Beziehung wird über die Tabelle bestellpositionen hergestellt.

4. Physisches Schema

Tables abrufen auf dem Pi per SSH

Das DB-Programm öffnen:

```
Tim@piroman:~ $ sudo mariadb
```

In die richtige DB gehen:

```
MariaDB [(none)]> USE bestellsystem;
```

Table: kunden

```
MariaDB [bestellsystem]> DESCRIBE kunden;
```

```
+------------------+-------------+------+-----+---------+----------------+
| Field            | Type        | Null | Key | Default | Extra          |
+------------------+-------------+------+-----+---------+----------------+
| id               | int(11)     | NO   | PRI | NULL    | auto_increment |
| firma            | varchar(100)| YES  |     | NULL    |                |
| vorname          | varchar(50) | YES  |     | NULL    |                |
| nachname         | varchar(50) | YES  |     | NULL    |                |
| email            | varchar(100)| YES  | UNI | NULL    |                |
| password         | varchar(255)| YES  |     | NULL    |                |
| adresse          | text        | YES  |     | NULL    |                |
| telefon_geschaeft| varchar(50) | YES  |     | NULL    |                |
| telefon_mobil    | varchar(50) | YES  |     | NULL    |                |
+------------------+-------------+------+-----+---------+----------------+
```

Table: admin

```
MariaDB [bestellsystem]> DESCRIBE admin;
```

```
+-------------+-------------+------+-----+---------+----------------+
| Field       | Type        | Null | Key | Default | Extra          |
+-------------+-------------+------+-----+---------+----------------+
| id          | int(11)     | NO   | PRI | NULL    | auto_increment |
| benutzername| varchar(50) | YES  | UNI | NULL    |                |
| password    | varchar(255)| NO   |     | NULL    |                |
+-------------+-------------+------+-----+---------+----------------+
```

Table: artikel

```
MariaDB [bestellsystem]> DESCRIBE artikel;
```

```
+-------------+--------------+------+-----+---------+----------------+
| Field       | Type         | Null | Key | Default | Extra          |
+-------------+--------------+------+-----+---------+----------------+
| id          | int(11)      | NO   | PRI | NULL    | auto_increment |
| name        | varchar(100) | NO   |     | NULL    |                |
| beschreibung| text         | YES  |     | NULL    |                |
| preis       | decimal(10,2)| NO   |     | NULL    |                |
+-------------+--------------+------+-----+---------+----------------+
```

Table: bestellungen

```
MariaDB [bestellsystem]> DESCRIBE bestellungen;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
kunde_id	int(11)	YES	MUL	NULL	
datum	datetime	YES		current_timestamp()	
artikel	varchar(100)	YES		NULL	
menge	int(11)	YES		NULL	
einzelpreis	decimal(10,2)	YES		NULL	
bezahlt	tinyint(1)	YES		0	
erstellt_am	datetime	YES		current_timestamp()	

Table: bestellpositionen

```
MariaDB [bestellsystem]> DESCRIBE bestellposition;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
bestellung_id	int(11)	NO	MUL	NULL	
artikel_id	int(11)	NO	MUL	NULL	
menge	int(11)	NO		NULL	

5. Normalisierung

1. Normalform (1NF): Jede Zelle enthält nur einen Wert, keine Listen oder mehrfachen Informationen.

2. Normalform (2NF): Alle Spalten hängen vollständig vom Primärschlüssel ab. Da nur einfache IDs als PKs verwendet werden, ist diese erfüllt.

3. Normalform (3NF): Keine redundanten Daten. Aktuell gibt es in bestellungen noch die Spalten artikel, menge, einzelpreis, die redundant zu bestellpositionen sind. Empfehlung: entfernen.

BCNF: Nach Entfernung der redundanten Spalten erfüllt.

6. Integritätsregeln & Constraints

- **NOT NULL** für Pflichtfelder wie kunden.email und admin.benutzername empfohlen.
- **UNIQUE** für eindeutige Felder (email in kunden, benutzername in admin).
- **CHECK**-Constraints für Wertebereiche (z. B. menge > 0, preis >= 0).
- **ON DELETE CASCADE** für abhängige Datensätze in bestellpositionen.
- **RESTRICT** oder **SET NULL** bei kunde_id in bestellungen, abhängig vom gewünschten Verhalten bei Kundendatenlöschung.

7. Indizes & Performance

- **Vorhanden:** UNIQUE-Indizes auf email und benutzername; FK-Indizes auf bestellung_id, artikel_id und kunde_id.
- **Empfehlung:** Zusätzliche Indizes für häufige Abfragen, z. B. bestellungen(datum).
- Nutzung von EXPLAIN für Analyse von Query-Performance.

8. Sicherheitskonzept

- Passwörter werden mit sicheren Hash-Algorithmen gespeichert.
- Zugriff auf die Datenbank nur mit minimal nötigen Rechten (Least Privilege).
- SQL-Injection wird durch Prepared Statements verhindert.
- Übertragung sensibler Daten erfolgt über HTTPS.
- Admin-Login ist gesondert abgesichert.