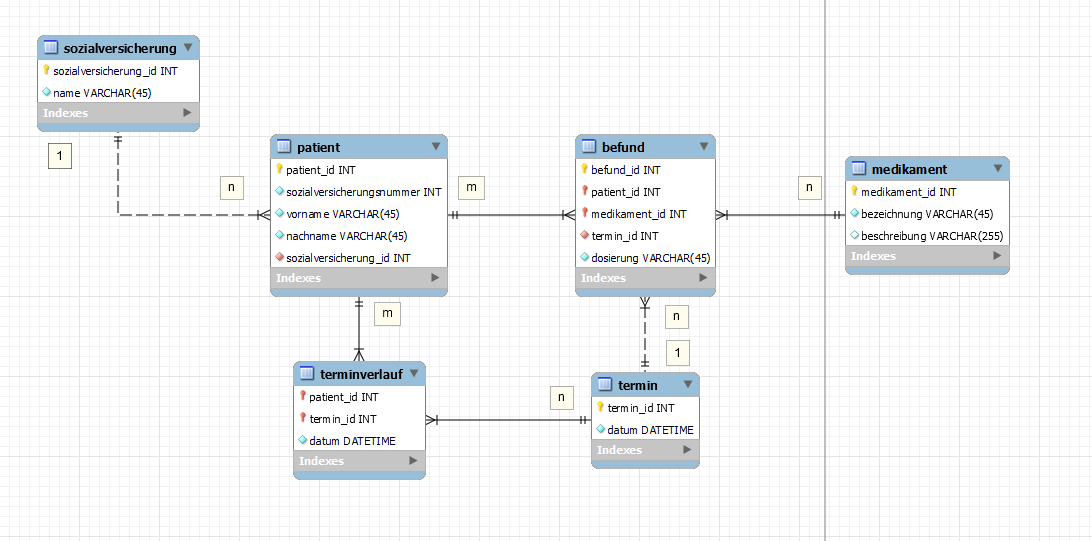
**ER-Diagramm:**

****

**SQL Script:**

-- MySQL Script generated by MySQL Workbench

-- Wed Aug 31 10:05:44 2022

-- Model: New Model Version: 1.0

-- MySQL Workbench Forward Engineering

SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0;

SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0;

SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION';

-- -----------------------------------------------------

-- Schema patientenverwaltung

-- -----------------------------------------------------

-- -----------------------------------------------------

-- Schema patientenverwaltung

-- -----------------------------------------------------

CREATE SCHEMA IF NOT EXISTS `patientenverwaltung` DEFAULT CHARACTER SET utf8mb4 ;

USE `patientenverwaltung` ;

-- -----------------------------------------------------

-- Table `patientenverwaltung`.`sozialversicherung`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `patientenverwaltung`.`sozialversicherung` (

`sozialversicherung\_id` INT NOT NULL AUTO\_INCREMENT,

`name` VARCHAR(45) NOT NULL,

PRIMARY KEY (`sozialversicherung\_id`),

UNIQUE INDEX `name\_UNIQUE` (`name` ASC))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `patientenverwaltung`.`patient`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `patientenverwaltung`.`patient` (

`patient\_id` INT NOT NULL AUTO\_INCREMENT,

`sozialversicherungsnummer` INT UNSIGNED NOT NULL,

`vorname` VARCHAR(45) NOT NULL,

`nachname` VARCHAR(45) NOT NULL,

`sozialversicherung\_id` INT NOT NULL,

PRIMARY KEY (`patient\_id`),

UNIQUE INDEX `sozialversicherungsnummer\_UNIQUE` (`sozialversicherungsnummer` ASC),

INDEX `fk\_patient\_sozialversicherung\_idx` (`sozialversicherung\_id` ASC),

CONSTRAINT `fk\_patient\_sozialversicherung`

FOREIGN KEY (`sozialversicherung\_id`)

REFERENCES `patientenverwaltung`.`sozialversicherung` (`sozialversicherung\_id`)

ON DELETE RESTRICT

ON UPDATE CASCADE)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `patientenverwaltung`.`medikament`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `patientenverwaltung`.`medikament` (

`medikament\_id` INT NOT NULL AUTO\_INCREMENT,

`bezeichnung` VARCHAR(45) NOT NULL,

`beschreibung` VARCHAR(255) NULL,

PRIMARY KEY (`medikament\_id`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `patientenverwaltung`.`termin`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `patientenverwaltung`.`termin` (

`termin\_id` INT NOT NULL AUTO\_INCREMENT,

`datum` DATETIME NOT NULL,

PRIMARY KEY (`termin\_id`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `patientenverwaltung`.`befund`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `patientenverwaltung`.`befund` (

`befund\_id` INT NOT NULL,

`patient\_id` INT NOT NULL,

`medikament\_id` INT NOT NULL,

`termin\_id` INT NOT NULL,

`dosierung` VARCHAR(45) NOT NULL,

PRIMARY KEY (`befund\_id`, `patient\_id`, `medikament\_id`),

INDEX `fk\_patient\_has\_medikament\_medikament1\_idx` (`medikament\_id` ASC),

INDEX `fk\_patient\_has\_medikament\_patient1\_idx` (`patient\_id` ASC),

INDEX `fk\_befund\_termin1\_idx` (`termin\_id` ASC),

CONSTRAINT `fk\_patient\_has\_medikament\_patient1`

FOREIGN KEY (`patient\_id`)

REFERENCES `patientenverwaltung`.`patient` (`patient\_id`)

ON DELETE RESTRICT

ON UPDATE CASCADE,

CONSTRAINT `fk\_patient\_has\_medikament\_medikament1`

FOREIGN KEY (`medikament\_id`)

REFERENCES `patientenverwaltung`.`medikament` (`medikament\_id`)

ON DELETE RESTRICT

ON UPDATE CASCADE,

CONSTRAINT `fk\_befund\_termin1`

FOREIGN KEY (`termin\_id`)

REFERENCES `patientenverwaltung`.`termin` (`termin\_id`)

ON DELETE RESTRICT

ON UPDATE CASCADE)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `patientenverwaltung`.`terminverlauf`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `patientenverwaltung`.`terminverlauf` (

`patient\_id` INT NOT NULL,

`termin\_id` INT NOT NULL,

`datum` DATETIME NOT NULL,

PRIMARY KEY (`patient\_id`, `termin\_id`),

INDEX `fk\_patient\_has\_termin\_termin1\_idx` (`termin\_id` ASC),

INDEX `fk\_patient\_has\_termin\_patient1\_idx` (`patient\_id` ASC),

CONSTRAINT `fk\_patient\_has\_termin\_patient1`

FOREIGN KEY (`patient\_id`)

REFERENCES `patientenverwaltung`.`patient` (`patient\_id`)

ON DELETE RESTRICT

ON UPDATE CASCADE,

CONSTRAINT `fk\_patient\_has\_termin\_termin1`

FOREIGN KEY (`termin\_id`)

REFERENCES `patientenverwaltung`.`termin` (`termin\_id`)

ON DELETE RESTRICT

ON UPDATE CASCADE)

ENGINE = InnoDB;

SET SQL\_MODE=@OLD\_SQL\_MODE;

SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS;

SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS;

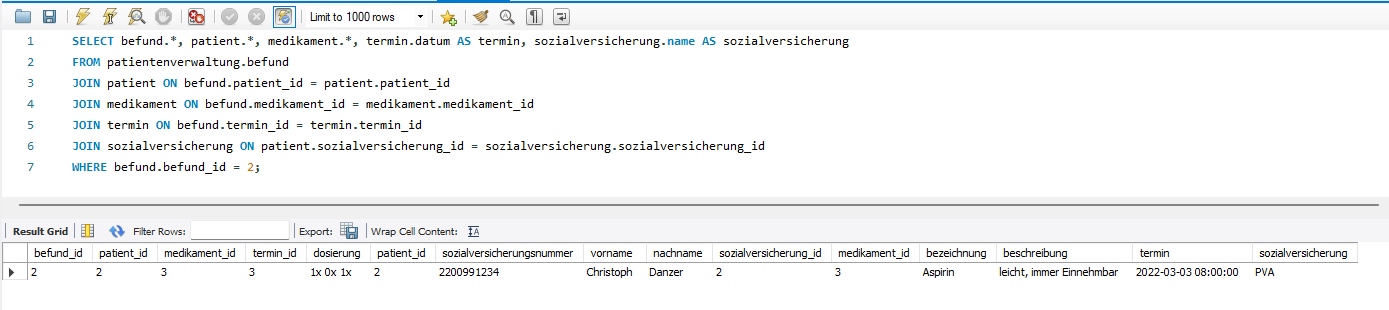
**Tabellen:**

| **Table** | **Angabe(Punkt)** | **Begründung** |
| --- | --- | --- |
| Patient |  | eigener Table um zukünftige Erweiterungen zu ermöglichen |
| sozialversicherung |  | 1:n zu Patient weil 1 Sozialversicherung zur mehreren Patienten gehören kann |
| Medikament |  | eigener Table um zukünftige Erweiterungen zu ermöglichen Beziehung= n:m zu Patient |
| Befund |  | Zwischentable von Patient zu Medikament |
| Termin |  | Eigener Table für saubere Organsierung |
| Terminverlauf |  | Zwischentable von Patient und Termin für korrekte neue Terminorganisierung |

**Foreign Keys:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Foreign Key name** | **betroffene Id** | **ON UPDATE** | | **GRUND** | | **ON DELETE** | | **GRUND2** |
| fk\_patient\_sozialversicherung | Sozialversicherung\_id | CASCADE | | Wenn eine neue Sozialversicherung hinterlegt wird | | RESTRICT | | Die Sozialversicherung kann nicht einfach gelöscht werden – Prüfung wieso gelöscht werden will! |
| fk\_patient\_has\_medikament\_patient1 | Patient\_id | CASCADE | | Wenn sich die ID des Patienten ändert soll diese auch im Befund geändert werden können | | RESTRICT | | Beim Löschen eines Patienten darf nicht sofort der Befund auch gelöscht werden |
| fk\_patient\_has\_medikament\_medikament1 | Medikament\_id | CASCADE | | Wenn sich im Befund die Medikamenten id ändert soll dies berücksichtigtwerden | | RESTRICT | | Löschen nur bedingt erlaubt -> Überprüfung ob Medikament noch in Benutzung -> Medikamenten Historie gehört aufgezeichnet |
| fk\_patient\_has\_termin\_patient1 | Patient\_id | CASCADE | | Für einfache organisierung, wenn sich Patient ändert | | RESTRICT | | Wenn gelöscht wird gehört gecheckt, ob noch Terminverlauf offen |
| fk\_patient\_has\_termin\_termin1 | Termin\_id | CASCADE | | Für einfache organisierung, wenn sich Patient ändert | | CASCADE RESTRICT | | Wenn gelöscht wird gehört gecheckt, ob noch Terminverlauf offen |
| fk\_befund\_termin1 | Termin\_id | CASCADE | Wenn sich der Termin ändert, soll dieser im Befund aktualisiert werden | | RESTRICT | | Das löschen des Termins ist nur erlaubt, wenn ein anderer Termin eingetragen wird | |

**Tests:**



Falsches Datum = 0000.00.00 in DB

