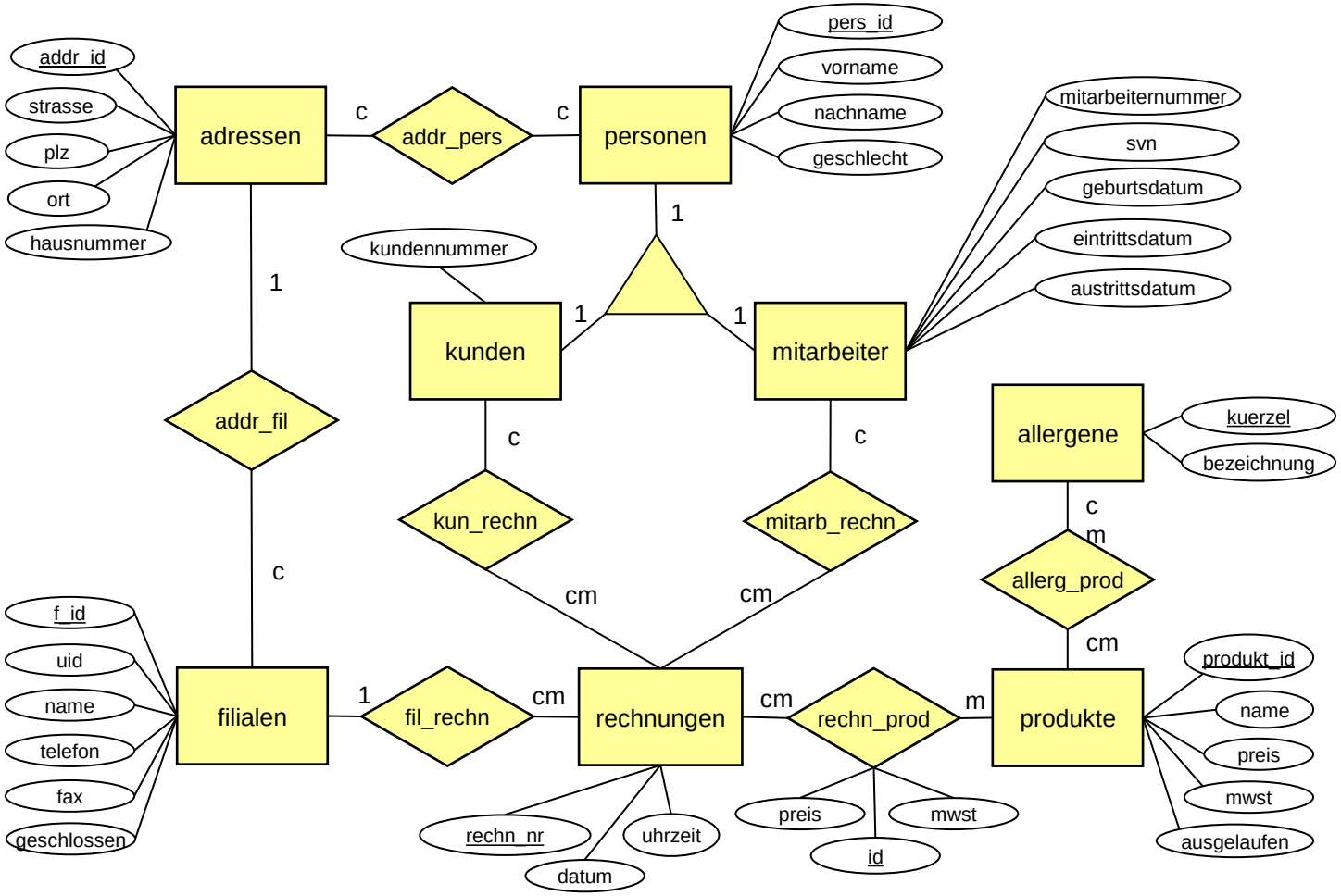
Übung - DDL - Toprast

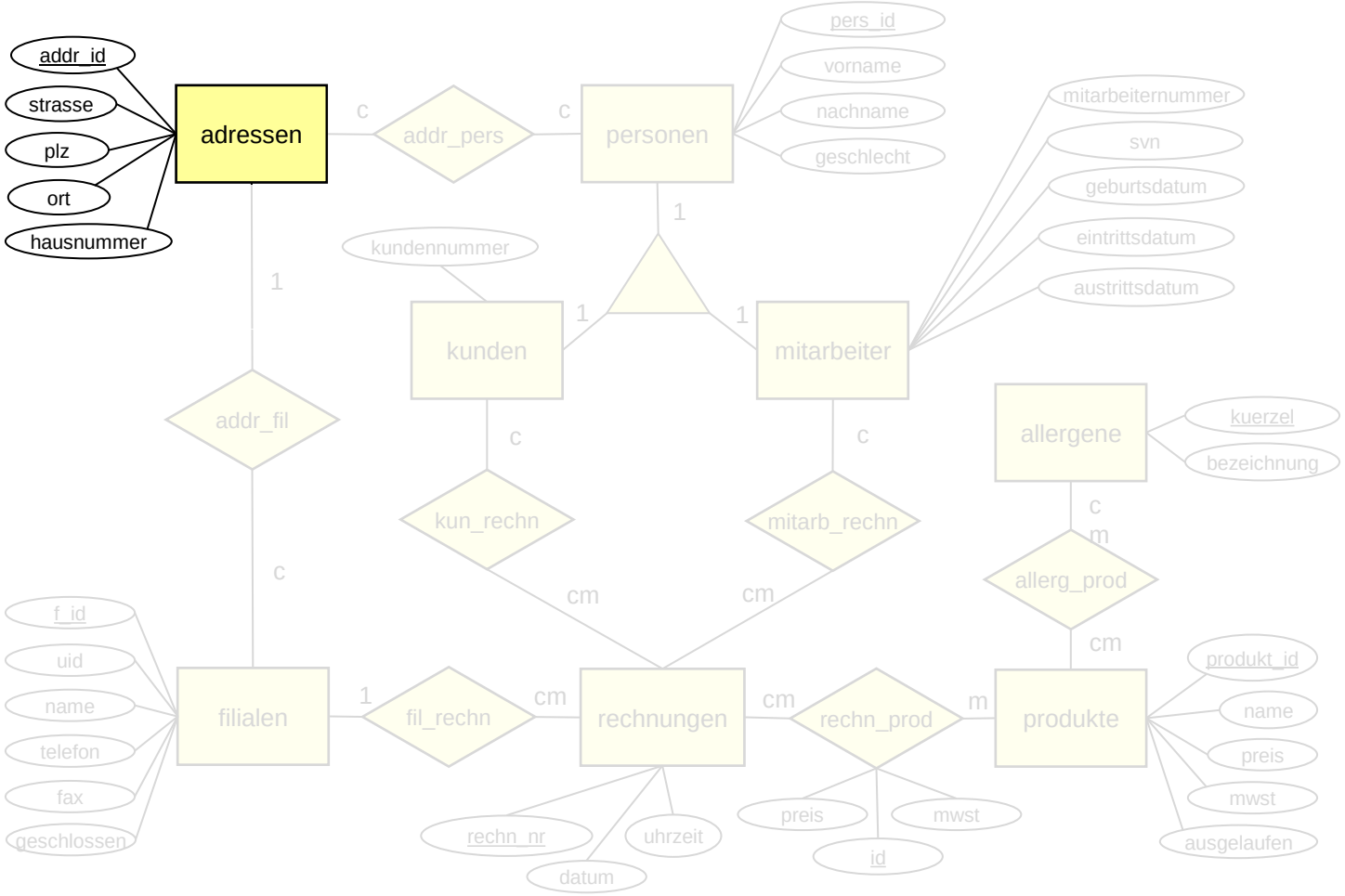
Martin Docsek

FH University of Applied Sciences
TECHNIKUM WIEN

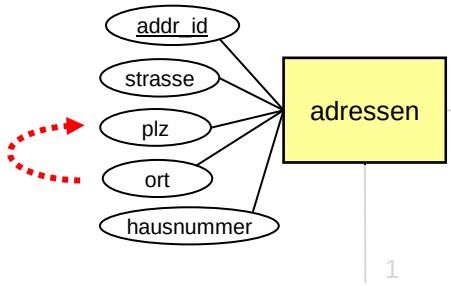
TOPRAST - ERM



TOPRAST – Umwandeln in Tabellen

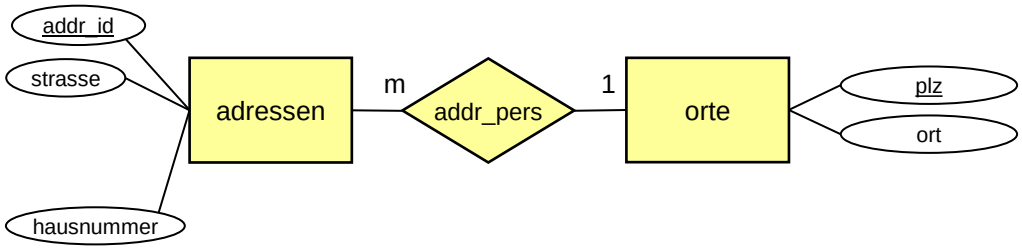


TOPRAST – Tabelle adressen



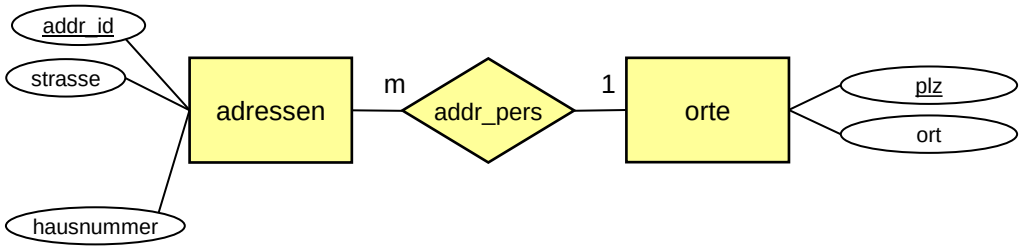
ort wird durch plz bestimmt
→ Verletzung der 3. NF

TOPRAST – Tabelle adressen



- **orte** (<u>plz</u>, ort)
- **adressen** (<u>addr_id</u>, strasse, hausnummer, #*plz*)

TOPRAST – Create Statements



Oracle

```
CREATE TABLE orte (
  plz INT NOT NULL,
  ort VARCHAR(64) NOT NULL,
  PRIMARY KEY (plz),
  CONSTRAINT ck_plz
    CHECK (plz between 1000 and 9999));

CREATE TABLE adressen (
  addr_id INT NOT NULL,
  strasse VARCHAR(64) NOT NULL,
  hausnummer VARCHAR(10) NULL,
  plz int NOT NULL,
  PRIMARY KEY (addr_id),
  CONSTRAINT fk_plz
    FOREIGN KEY (plz)
    REFERENCES orte (plz));
```

PostgreSQL

```
CREATE TABLE IF NOT EXISTS "orte" (
  "plz" INT NOT NULL,
  "ort" VARCHAR(64) NOT NULL,
  PRIMARY KEY ("plz"),
  CONSTRAINT "ck_plz"
    CHECK ("plz" between 1000 and 9999));

CREATE TABLE IF NOT EXISTS "adressen" (
  "addr_id" SERIAL NOT NULL,
  "strasse" VARCHAR(64) NOT NULL,
  "plz" int NOT NULL,
  "hausnummer" VARCHAR(10) NULL,
  PRIMARY KEY ("addr_id"),
  CONSTRAINT "fk_plz"
    FOREIGN KEY ("plz")
    REFERENCES "orte" ("plz"));
```

Erklärungen

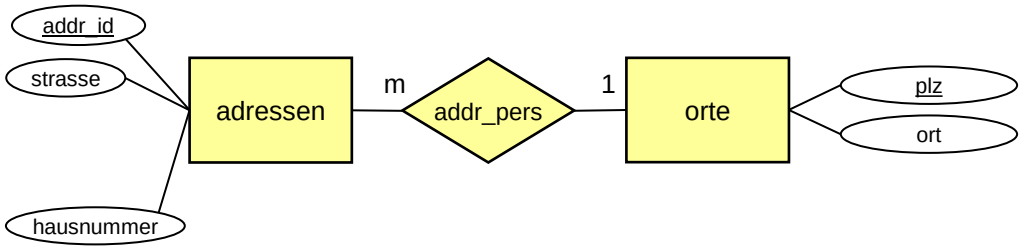
- `CREATE TABLE IF NOT EXISTS …`
  - Erweiterung in PostgreSQL (und anderen DBS)
  - prüft, ob die Tabelle schon existiert, um eine Fehlermeldung zu verhindern
  - in Oracle gibt es diese Erweiterung nicht
    - hier kann das über PL/SQL Blöcke umgesetzt werden

- Constraints
  - NOT NULL
    - dieser Constraint muss als Column Constraint (inline) definiert werden
  - PRIMARY KEY, UNIQUE, FOREIGN KEY, CHECK
    - können als Column Constraint (inline) oder Table Constraint (outline) definiert werden
    - Ausnahmen: setzen sich PRIMARY KEY, UNIQUE und FOREIGN KEY aus mehreren Attributen zusammen, müssen sie als Table Constraint (outline) definiert werden.

Erklärungen

- `"addr_id" SERIAL NOT NULL …`
  - SERIAL ist ein Autoincrement Integer Datentyp in PostgreSQL
  - zum automatischen Erzeugen von Integerwerten
  - In Oracle gibt es keinen vergleichbaren Datentyp
    - Alternativen:
      - Sequence zum Erzeugen fortlaufender Werte
      - ab Version 12c kann eine Spalte für fortlaufenden Nummerierung als IDENTITY definiert werden
      - Trigger

TOPRAST – Beispiele Constraints



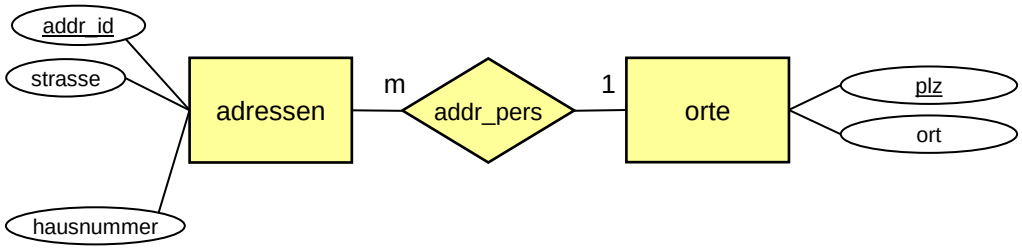
Column Constraints (inline)

```
CREATE TABLE orte (
  plz INT PRIMARY KEY CONSTRAINT ck_plz CHECK (plz between 1000 and 9999),
  ort VARCHAR(64) NOT NULL);

CREATE TABLE adressen (
  addr_id INT PRIMARY KEY ,
  strasse VARCHAR(64) NOT NULL,
  hausnummer VARCHAR(10) NULL,
  plz int NOT NULL CONSTRAINT fk_plz REFERENCES orte (plz));
```

Table Constraints (outline) außer NOT NULL

```
CREATE TABLE orte (
  plz INT NOT NULL,
  ort VARCHAR(64) NOT NULL,
  PRIMARY KEY (plz),
  CONSTRAINT ck_plz CHECK (plz between 1000 and 9999));

CREATE TABLE adressen (
  addr_id INT NOT NULL,
  strasse VARCHAR(64) NOT NULL,
  hausnummer VARCHAR(10) NULL,
  plz int NOT NULL,
  PRIMARY KEY (addr_id),
  CONSTRAINT fk_plz FOREIGN KEY (plz) REFERENCES orte (plz));
```

Case (in)sensitivity



Oracle

```
CREATE TABLE orte (
 "Plz" INT NOT NULL,
 "ort" VARCHAR(64) NOT NULL,
  PRIMARY KEY ("Plz"),
  CONSTRAINT ck_plz
    CHECK ("Plz" between 1000 and 9999));
```

PostgreSQL

```
CREATE TABLE IF NOT EXISTS orte (
  "Plz" INT NOT NULL,
  "ORT" VARCHAR(64) NOT NULL,
  PRIMARY KEY ("Plz"),
  CONSTRAINT "ck_plz"
    CHECK ("Plz" between 1000 and 9999));
```

Die Aussage „SQL ist case insensitive“ ist nicht voll umfänglich gültig. Die Behandlung der Groß- und Kleinschreibung ist in verschiedenen DBS unterschiedlich umgesetzt.

- Oracle wandelt Objektnamen in Großbuchstaben um
- PostgreSQL wandelt Objektnamen in Kleinbuchstaben um
- Bezeichnungen in Hochkommas (einfache und doppelte) werden case sensitive behandelt.

Case (in)sensitivity

SQL Statement	Oracle		PostgreSQL	
	Tabellenname	Attributsname	Tabellenname	Attributsname
CREATE TABLE person (nachname VARCHAR(100));	PERSON	NACHNAME	person	nachname
CREATE TABLE Person (Nachname VARCHAR(100));	PERSON	NACHNAME	person	nachname
CREATE TABLE PERSON (NACHNAME VARCHAR(100));	PERSON	NACHNAME	person	nachname
CREATE TABLE "person" ("nachname" VARCHAR(100));	person	nachname	person	nachname
CREATE TABLE "Person" ("Nachname" VARCHAR(100));	Person	Nachname	Person	Nachname
CREATE TABLE "PERSON" ("NACHNAME" VARCHAR(100));	PERSON	NACHNAME	PERSON	NACHNAME

Beispiel

```
create table Person (
  nachname varchar(100),
  "Nachname" varchar(100),
  "nachname" varchar(100));
```

in Oracle: PERSON (NACHNAME, Nachname, nachname)
in PostgreSQL: Fehlermeldung, da 2x nachname

Oracle - Select Statement:

```
SELECT nachname, "Nachname", "nachname"
FROM person;
```

Beispiel

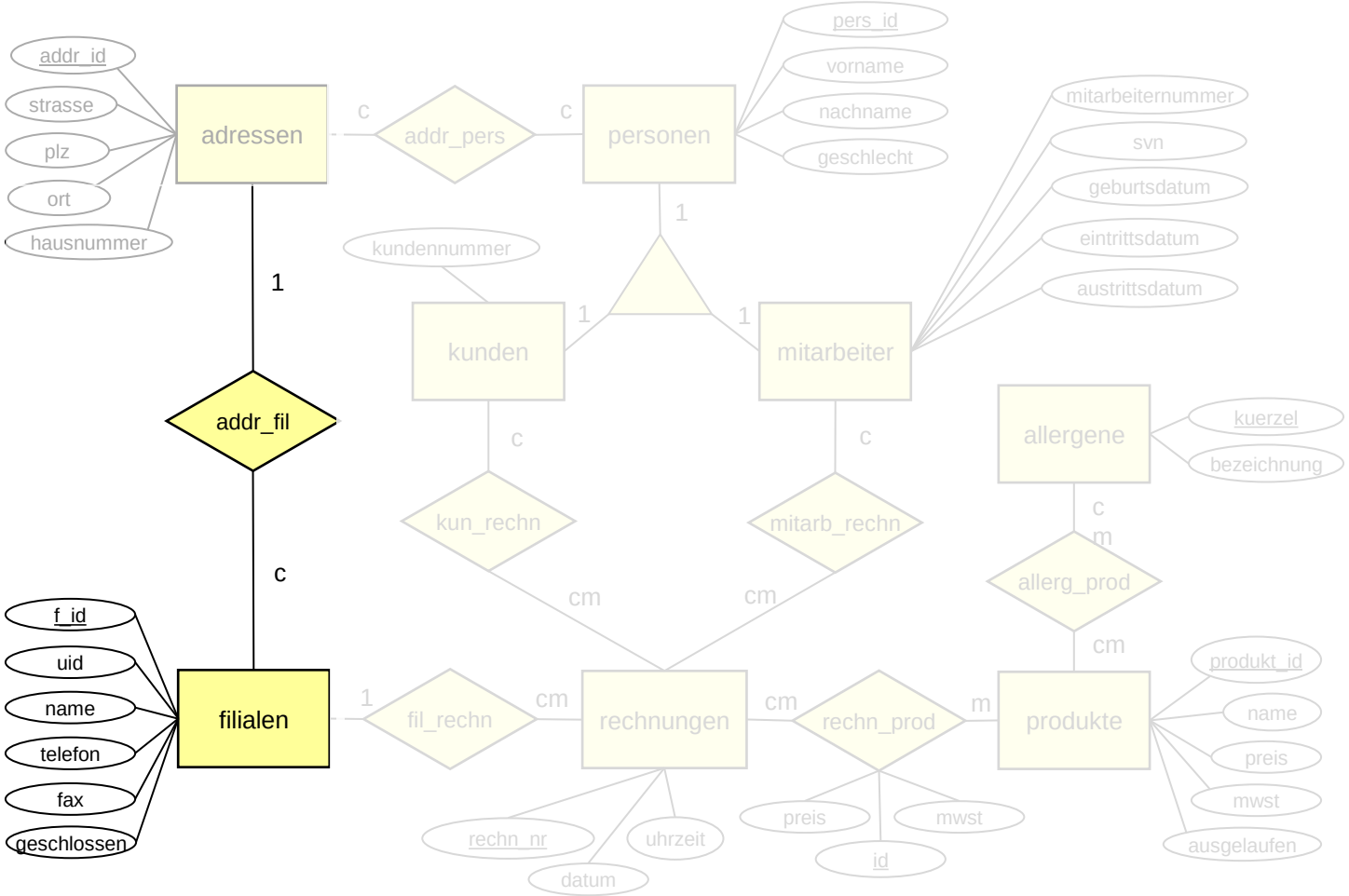
```
create table Person (
  NACHNAME varchar(100),
  "Nachname" varchar(100),
  "NACHNAME" varchar(100));
```

in Oracle: Fehlermeldung, da 2x NACHNAME
in PostgreSQL: PERSON (nachname, Nachname, NACHNAME)

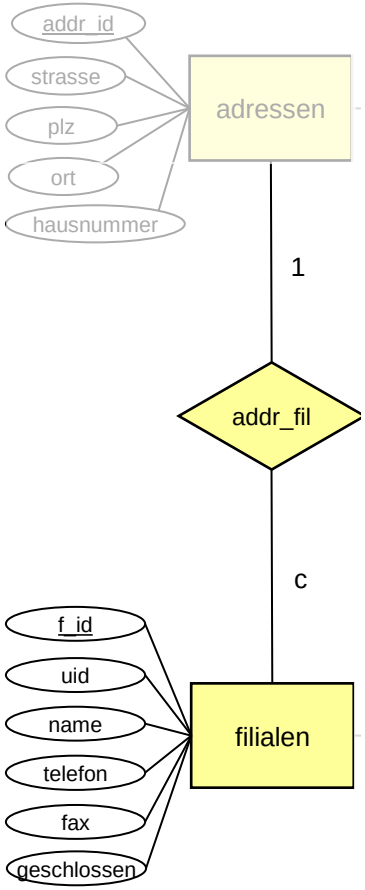
PostgreSQL - Select Statement:

```
SELECT nachname, "Nachname", "NACHNAME"
FROM person;
```

TOPRAST – Tabelle filialen

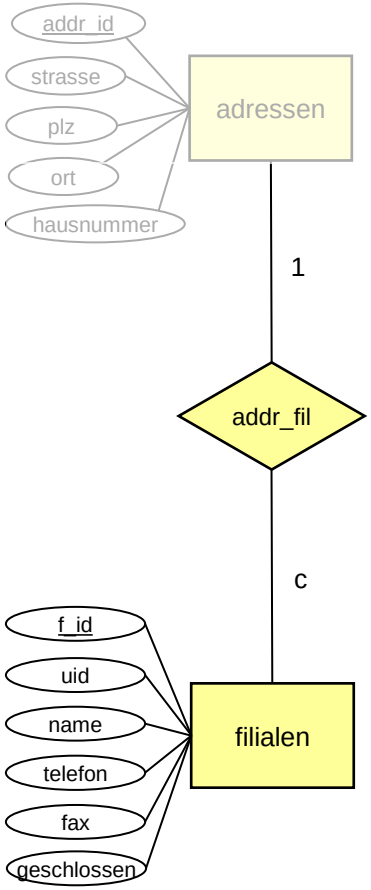


TOPRAST – Tabelle filialen



- **filialen** (<u>f_id</u>, uid, name, telefon, fax, geschlossen, *#fk_address_id*)

TOPRAST – Create Statement filialen

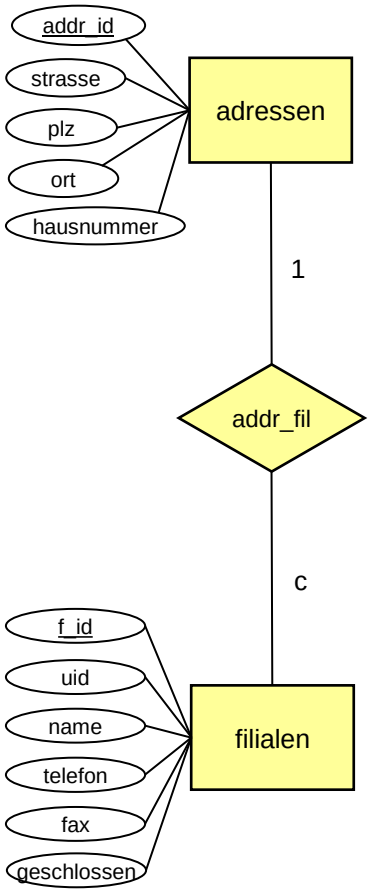


- **filialen** (<u>f_id</u>, uid, name, telefon, fax, geschlossen, *#fk_address_id*)

Oracle

```
CREATE TABLE filialen (
  f_id INT NOT NULL,
  "uid" VARCHAR(12) NULL,
  name VARCHAR(45) NOT NULL,
  telefon VARCHAR(45) NULL,
  fax VARCHAR(45) NULL,
  fk_addr_id INT NOT NULL,
  geschlossen CHAR(1) DEFAULT 0 NOT NULL,
  PRIMARY KEY (f_id),
  CONSTRAINT fk_filialen_adressen
    FOREIGN KEY (fk_addr_id)
    REFERENCES adressen (addr_id),
  CONSTRAINT ck_filialen_geschlossen
    CHECK (CAST(geschlossen AS INT) IN (0,1)));
```

TOPRAST – Create Statement filialen



- **filialen** (<u>f_id</u>, uid, name, telefon, fax, geschlossen, *#fk_address_id*)

PostgreSQL

```
CREATE TABLE IF NOT EXISTS filialen (
  f_id SERIAL NOT NULL,
  uid VARCHAR(12) NULL,
  name VARCHAR(45) NOT NULL,
  telefon VARCHAR(45) NULL,
  fax VARCHAR(45) NULL,
  fk_addr_id INT NOT NULL,
  geschlossen BOOLEAN NOT NULL DEFAULT FALSE,
  PRIMARY KEY (f_id),
  CONSTRAINT fk_filialen_adressen
    FOREIGN KEY (fk_addr_id)
    REFERENCES adressen (addr_id)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);
```

Erklärungen

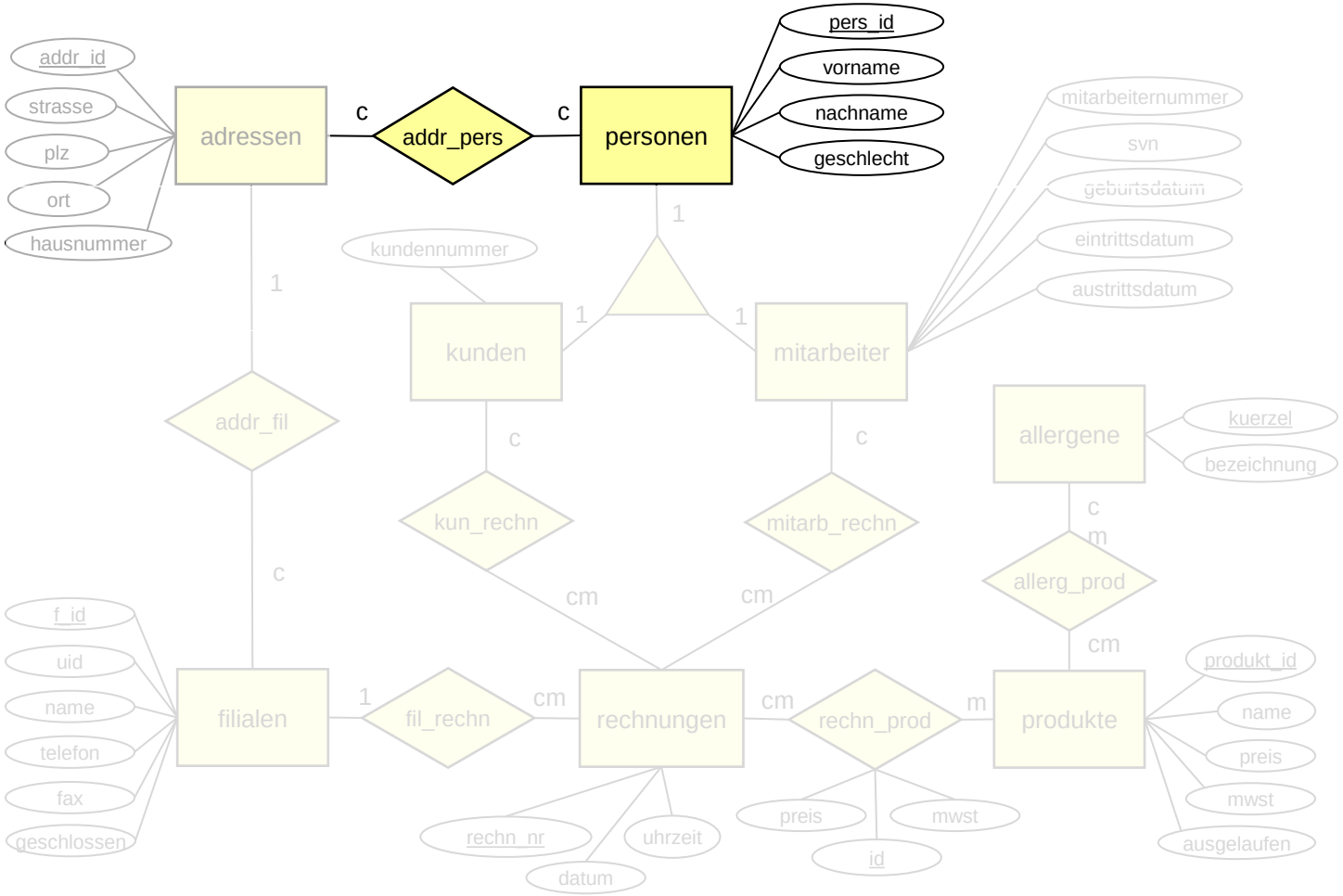
- … "uid" VARCHAR(12) NULL …
  - UID ist eine Oracle Variable (UserID des angemeldeten Benutzers)
  - muss in Oracle unter Anführungszeichen gesetzt werden
    - sonst Fehlermeldung in Create Statement oder falscher Wert in Select Statement

- …geschlossen CHAR(1) DEFAULT 0 NOT NULL…
  - in Oracle gibt es keinen Datentyp Boolean (kein ANSI Datentyp)
  - Defaultwerte müssen bei Oracle direkt nach dem Datentyp definiert werden

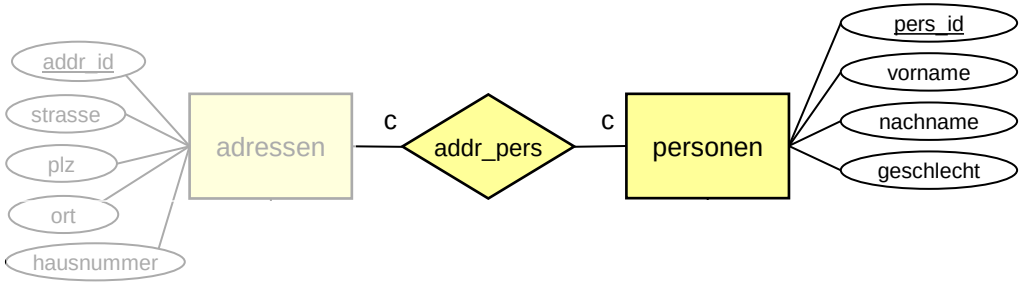
- … CHECK (CAST(geschlossen AS INT) IN (0,1)) …
  - CAST für Typumwandlung in Oracle hier nicht erforderlich (implizite Umwandlung)
  - Alternativen:
    - … CHECK (geschlossen IN (0,1)) …
    - … CHECK (geschlossen IN ('0', '1', 'f', 't')) …

- … ON DELETE NO ACTION ON UPDATE NO ACTION …
  - kann in PostgreSQL auch weggelassen werden
  - in Oracle nur ON DELETE SET NULL bzw. ON DELETE CASCADE
    - kein ON UPDATE bzw. NO ACTION

TOPRAST – Tabelle personen

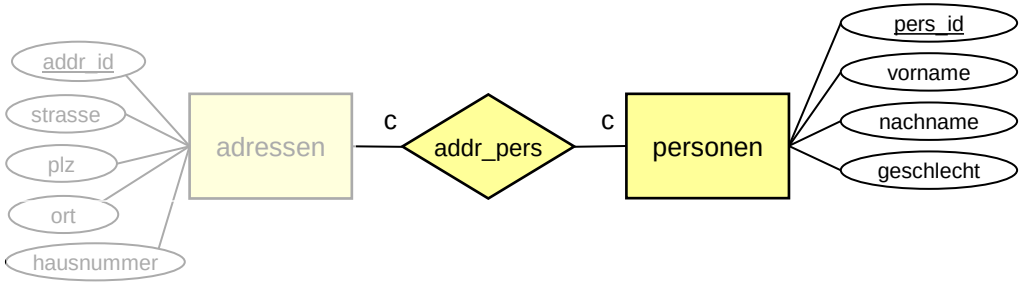


TOPRAST – Tabelle personen



- **personen** (pers_id, vorname, nachname, geschlecht, *#fk_address_id*)

TOPRAST – Create Statements personen



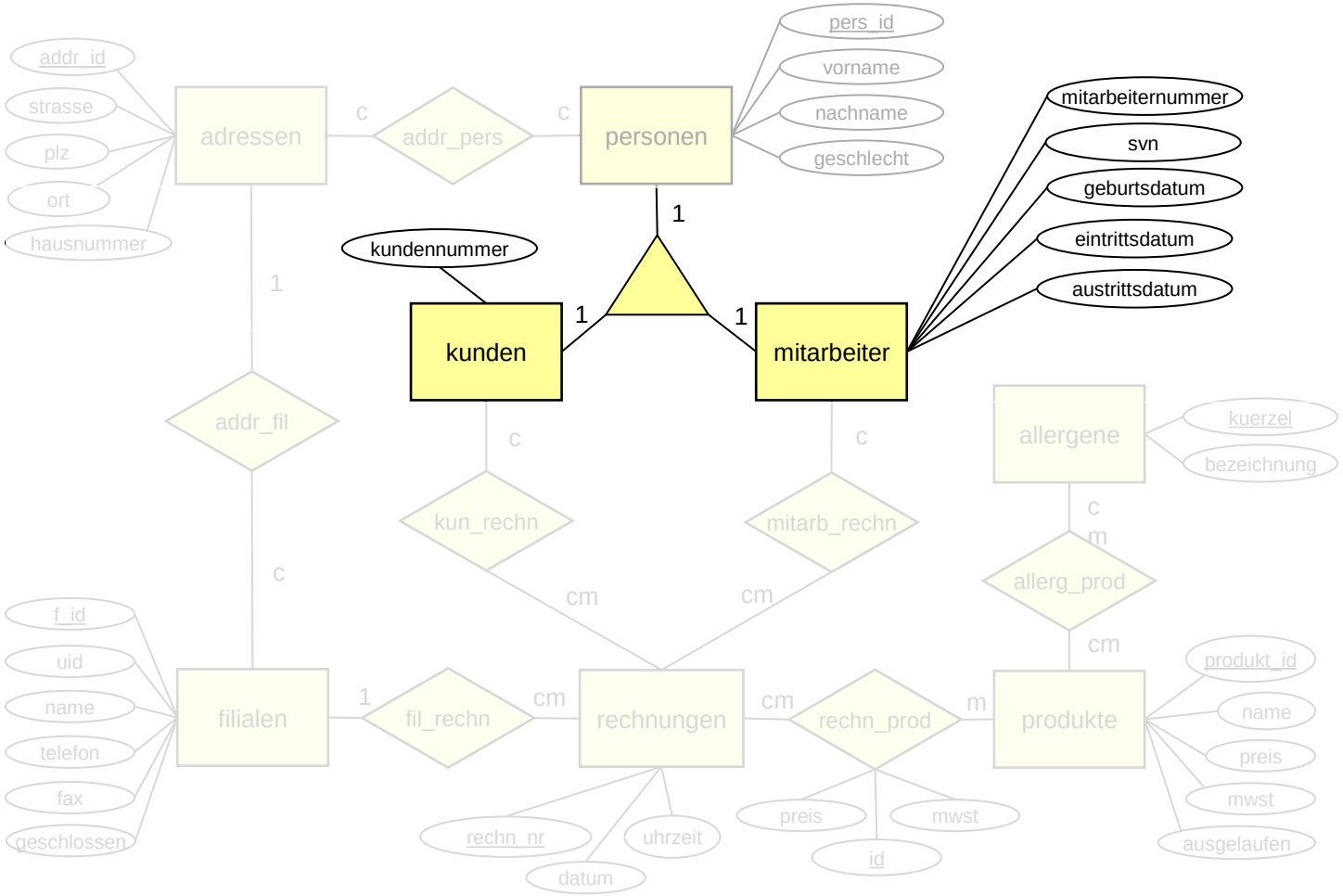
Oracle

```
CREATE TABLE personen (
  pers_id INT NOT NULL,
  vorname VARCHAR(45) NULL,
  nachname VARCHAR(45) NULL,
  geschlecht CHAR(1),
  fk_addr_id INT NULL,
  PRIMARY KEY (pers_id),
  CONSTRAINT ck_personen_geschlecht
    CHECK (geschlecht IN('m','w')),
  CONSTRAINT fk_personen_adressen
    FOREIGN KEY (fk_addr_id)
    REFERENCES adressen (addr_id)
    ON DELETE SET NULL);
```

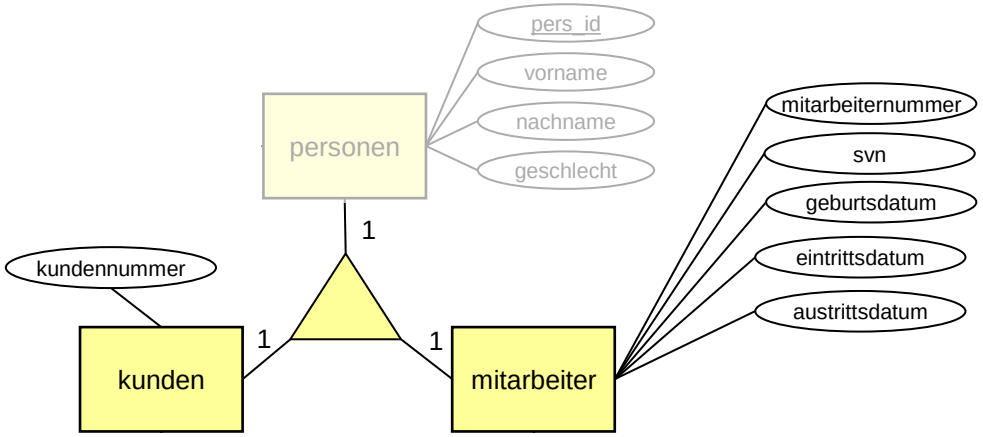
PostgreSQL

```
CREATE TABLE IF NOT EXISTS personen (
  pers_id SERIAL NOT NULL,
  vorname VARCHAR(45) NULL,
  nachname VARCHAR(45) NULL,
  geschlecht CHAR(1),
  fk_addr_id INT NULL,
  PRIMARY KEY (pers_id),
  CONSTRAINT ck_personen_geschlecht
    CHECK (geschlecht IN('m','w')),
  CONSTRAINT fk_personen_adressen
    FOREIGN KEY (fk_addr_id)
    REFERENCES adressen (addr_id)
    ON DELETE SET NULL
    ON UPDATE SET NULL);
```

TOPRAST – Tabellen kunden, mitarbeiter

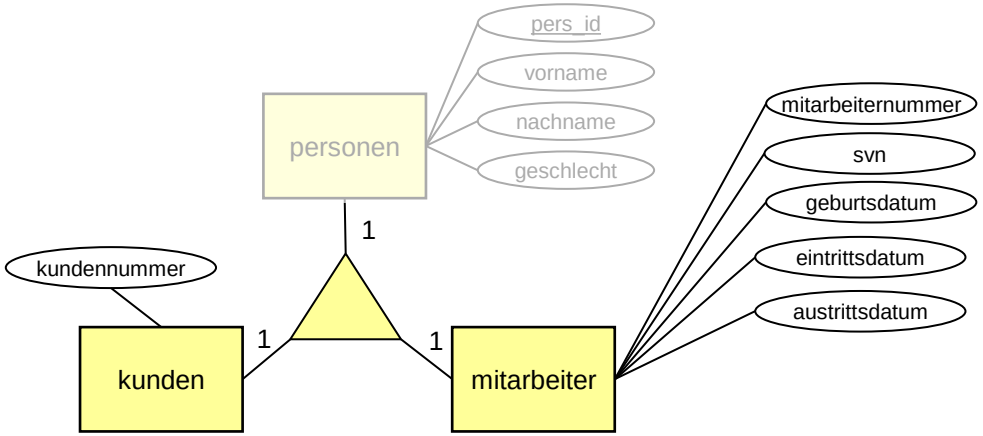


TOPRAST – Tabellen kunden, mitarbeiter



- **kunden** (#*pers_id*, kundennummer)
- **mitarbeiter** (#*pers_id*, mitarbeiternummer, svn, geburtsdatum, eintrittsdatum, austrittsdatum)

TOPRAST – Create Statements kunden



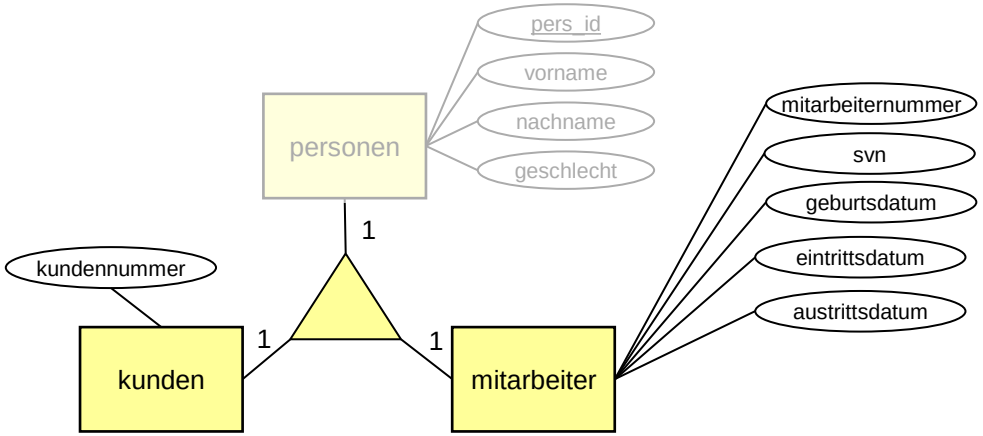
Oracle

```
CREATE TABLE kunden (
  pers_id INT NOT NULL,
  kundennummer INT UNIQUE NOT NULL,
  PRIMARY KEY (fk_pers_id),
  CONSTRAINT fk_kunden_personen
    FOREIGN KEY (fk_pers_id)
    REFERENCES personen (pers_id)
    ON DELETE CASCADE);
```

PostgreSQL

```
CREATE TABLE IF NOT EXISTS kunden (
  pers_id INT NOT NULL,
  kundennummer INT UNIQUE NOT NULL,
  PRIMARY KEY (fk_pers_id),
  CONSTRAINT fk_kunden_personen
    FOREIGN KEY (fk_pers_id)
    REFERENCES personen (pers_id)
    ON DELETE CASCADE
    ON UPDATE CASCADE);
```

TOPRAST – Create Statements mitarbeiter



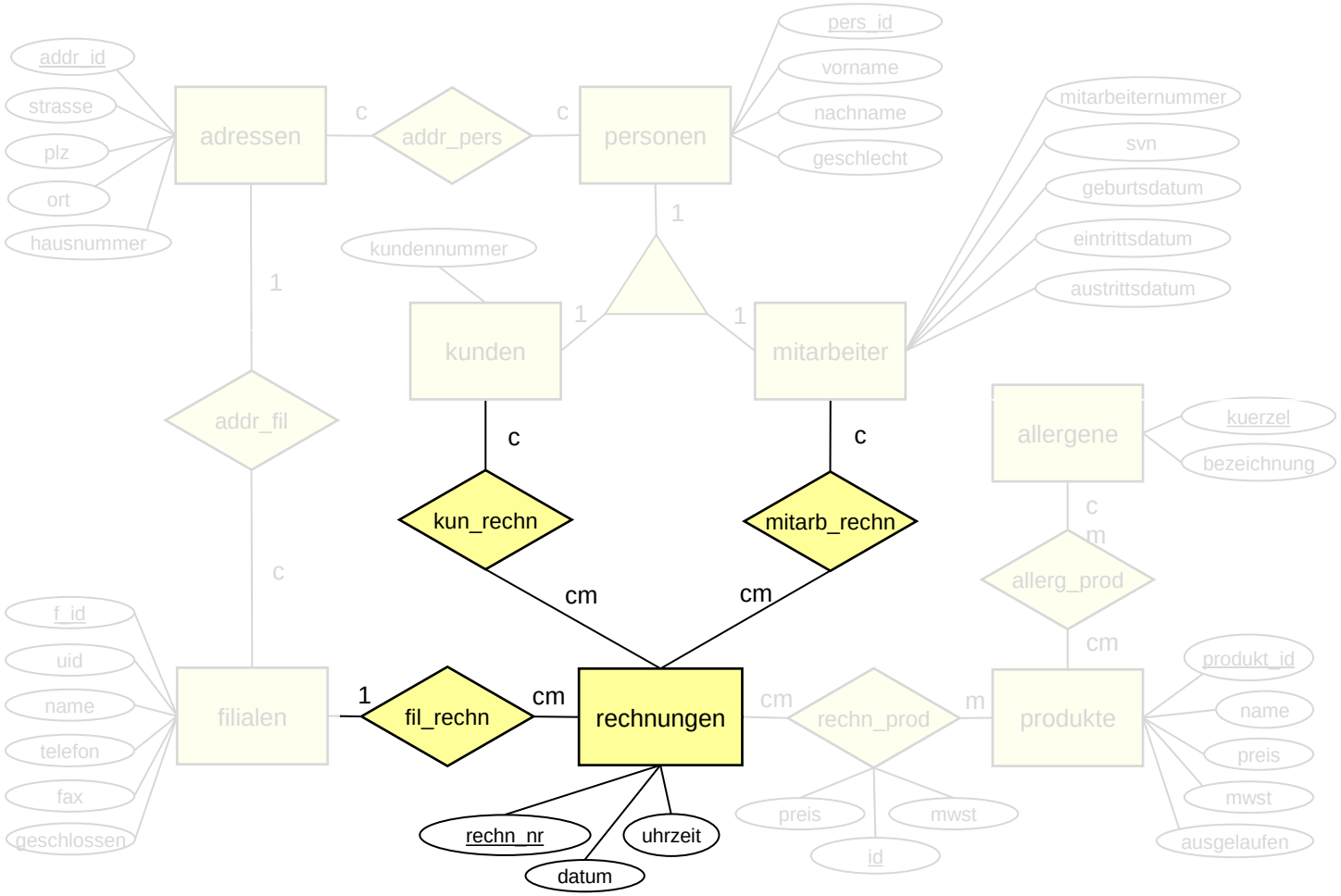
Oracle

```
CREATE TABLE mitarbeiter (
  pers_id INT NOT NULL,
  mitarbeiternummer INT UNIQUE NOT NULL,
  svn INT NOT NULL,
  geburtsdatum DATE NOT NULL,
  eintrittsdatum DATE NOT NULL,
  austrittsdatum DATE NULL,
  PRIMARY KEY (fk_pers_id),
  CONSTRAINT ck_mitarbeiter_svn
    CHECK (svn BETWEEN 1000 AND 9999),
  CONSTRAINT fk_mitarbeiter_personen
    FOREIGN KEY (fk_pers_id)
    REFERENCES personen (pers_id));
```

PostgreSQL

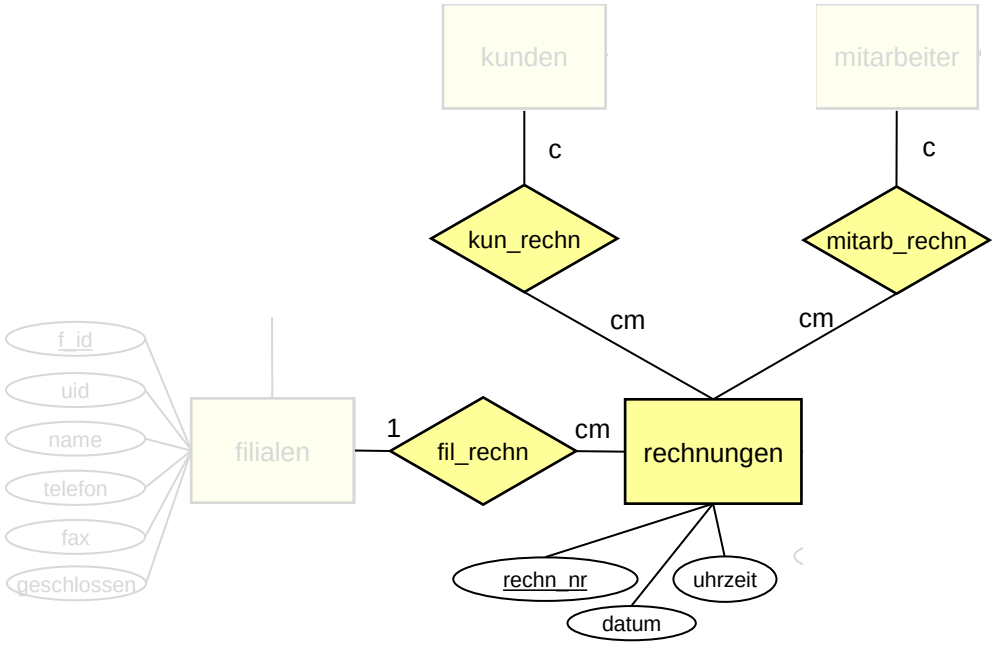
```
CREATE TABLE IF NOT EXISTS mitarbeiter (
  pers_id INT NOT NULL,
  mitarbeiternummer INT UNIQUE NOT NULL,
  svn INT NOT NULL,
  geburtsdatum DATE NOT NULL,
  eintrittsdatum DATE NOT NULL,
  austrittsdatum DATE NULL,
  PRIMARY KEY (fk_pers_id),
  CONSTRAINT ck_svn
    CHECK (svn BETWEEN 1000 AND 9999),
  CONSTRAINT fk_mitarbeiter_personen
    FOREIGN KEY (fk_pers_id)
    REFERENCES personen (pers_id));
```

TOPRAST – Tabelle rechnungen



TOPRAST – Tabelle rechnungen

- **rechnungen** (<u>rechn_nr</u>, datum, uhrzeit, *#fk_f_id*, *#fk_kunden_pers_id*, *#fk_mitarbeiter_pers_id*)



TOPRAST – Create Statements rechnungen

Oracle

```
CREATE TABLE rechnungen (
  rechn_nr INT NOT NULL,
  datum DATE NOT NULL,
  fk_f_id INT NOT NULL,
  fk_kunden_pers_id INT NULL,
  fk_mitarbeiter_pers_id INT NULL,
  PRIMARY KEY (rechn_nr),
  CONSTRAINT fk_rechnungen_filialen
    FOREIGN KEY (fk_f_id)
    REFERENCES filialen (f_id),
  CONSTRAINT fk_rechnungen_kunden
    FOREIGN KEY (fk_kunden_pers_id)
    REFERENCES kunden (fk_pers_id)
    ON DELETE SET NULL,
  CONSTRAINT fk_rechnungen_mitarbeiter
    FOREIGN KEY (fk_mitarbeiter_pers_id)
    REFERENCES mitarbeiter (fk_pers_id)
    ON DELETE SET NULL);
```

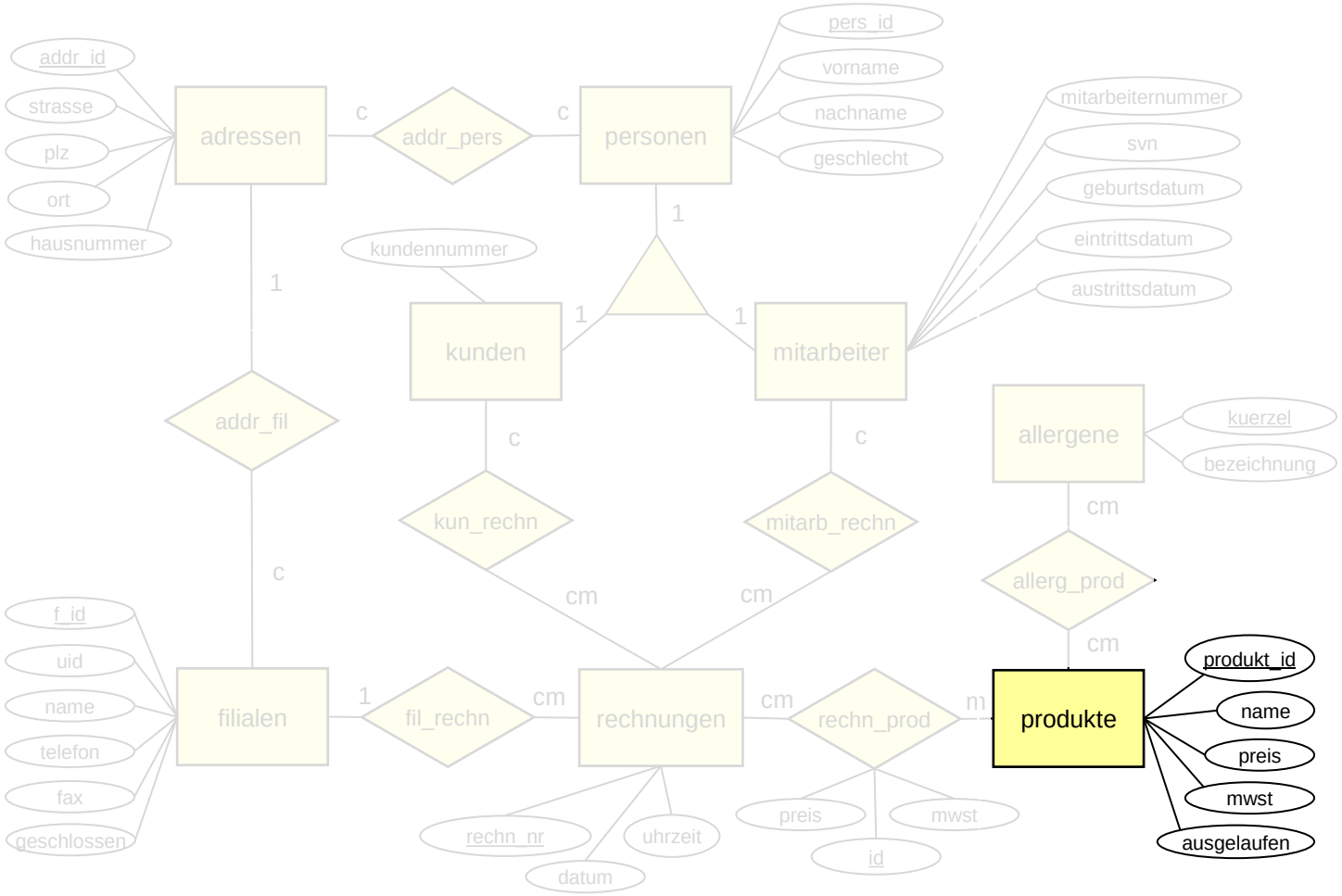
PostgreSQL

```
CREATE TABLE IF NOT EXISTS rechnungen (
  rechn_nr SERIAL NOT NULL,
  datum DATE NOT NULL,
  uhrzeit TIME NOT NULL,
  fk_f_id INT NOT NULL,
  fk_kunden_pers_id INT NULL,
  fk_mitarbeiter_pers_id INT NULL,
  PRIMARY KEY (rechn_nr),
  CONSTRAINT fk_rechnungen_filialen
    FOREIGN KEY (fk_f_id)
    REFERENCES filialen (f_id)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT fk_rechnungen_kunden
    FOREIGN KEY (fk_kunden_pers_id)
    REFERENCES kunden (fk_pers_id)
    ON DELETE SET NULL
    ON UPDATE SET NULL,
  CONSTRAINT fk_rechnungen_mitarbeiter
    FOREIGN KEY (fk_mitarbeiter_pers_id)
    REFERENCES mitarbeiter (fk_pers_id)
    ON DELETE SET NULL
    ON UPDATE SET NULL);
```

Erklärungen

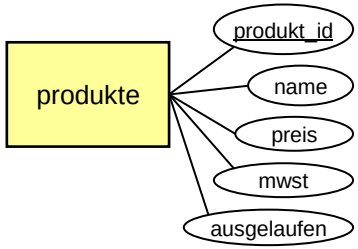
- Datentyp Date
  - Oracle: speichert Datum + Uhrzeit (dd-mm-yyyy hh:mm:ss)
  - PostgreSQL: speichert nur Datum (dd-mm-yyyy)
- Datentyp Time
  - Oracle: nicht verfügbar
  - PostgreSQL: speichert Zeit (hh:mm:ss)
- Datentyp Timestame
  - Oracle und PostgreSQL
  - speichert Datum + Zeit + Sekundenbruchteile (dd-mm-yyyy hh:mm:ss.ff)

TOPRAST –Tabelle produkte



TOPRAST – Create Statement produkte

- **produkte** (produkt_id, name, preis, mwst, ausgelaufen)



TOPRAST – Create Statement produkte

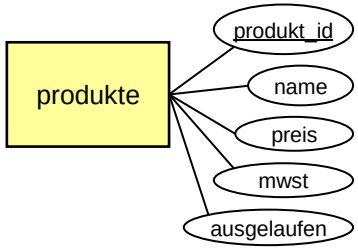
Oracle

```
CREATE TABLE produkte (
  produkt_id INT NOT NULL,
  name VARCHAR(64) NOT NULL,
  preis REAL NOT NULL,
  mwst REAL NOT NULL,
  ausgelaufen CHAR(1) DEFAULT 0 NOT NULL,
  PRIMARY KEY (produkt_id),
  CONSTRAINT ck_produkte_ausgelaufen
    CHECK (CAST(ausgelaufen AS INT) IN (0,1)));
```

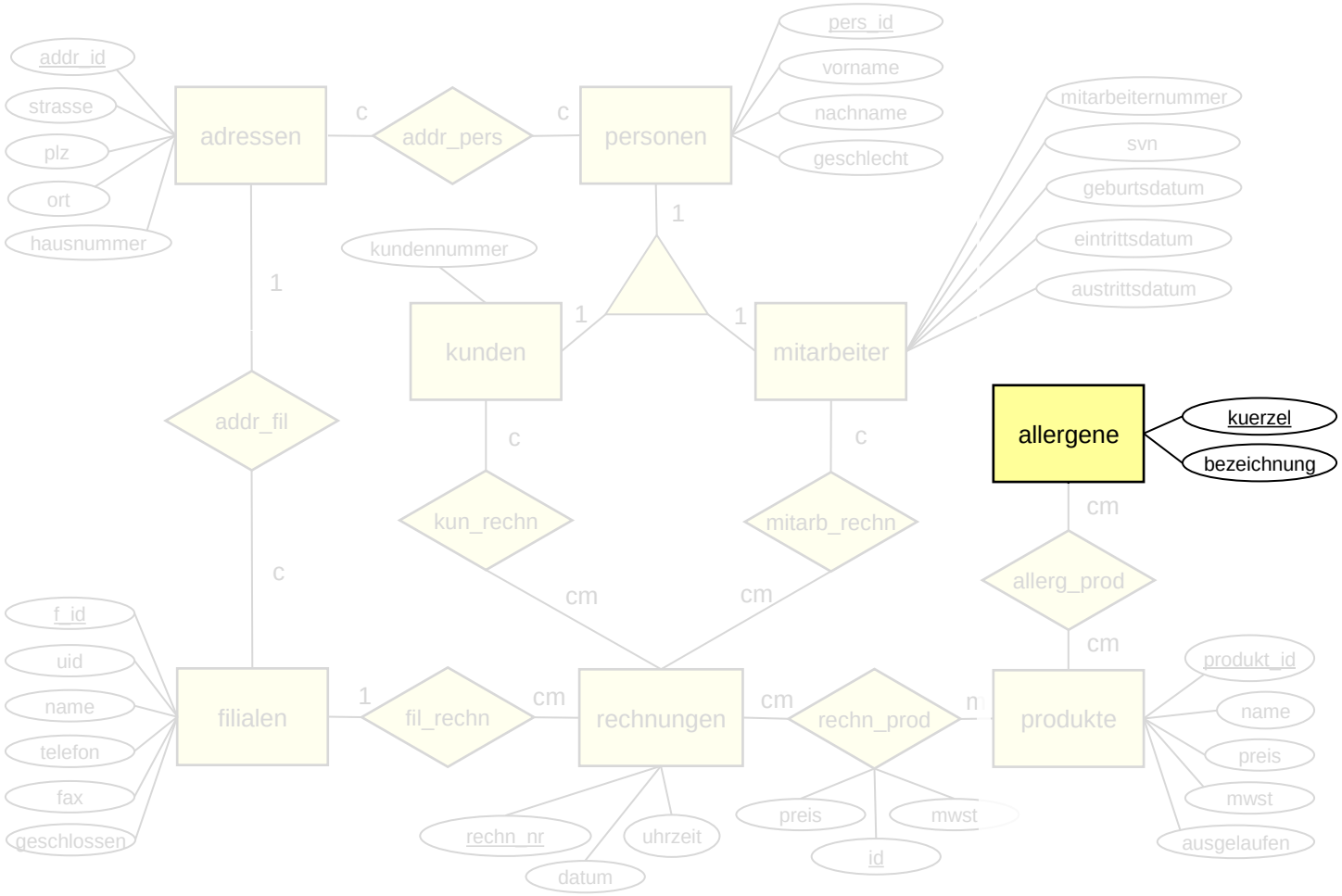
PostgreSQL

```
CREATE TABLE IF NOT EXISTS produkte (
  produkt_id SERIAL NOT NULL,
  name VARCHAR(64) NOT NULL,
  preis REAL NOT NULL,
  mwst REAL NOT NULL,
  ausgelaufen BOOLEAN NOT NULL DEFAULT FALSE,
  PRIMARY KEY (produkt_id));
```

- **produkte** (produkt_id, name, preis, mwst, ausgelaufen)

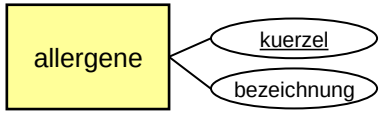


TOPRAST – Tabelle allergene



TOPRAST – Create Statement allergene

- **allergene** (<u>kuerzel</u>, bezeichnung)



TOPRAST – Create Statement allergene

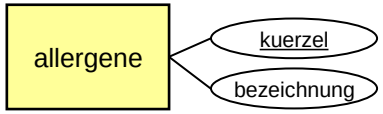
Oracle

```
CREATE TABLE allergene (
  kuerzel CHAR(1) NOT NULL,
  bezeichnung VARCHAR(64) NOT NULL,
  PRIMARY KEY (kuerzel),
  CONSTRAINT ck_allergene_kuerzel
    CHECK (kuerzel IN ('A', 'B', 'C', 'D', 'E',
         F','G','H','L','M','N','O','P','R')));
```

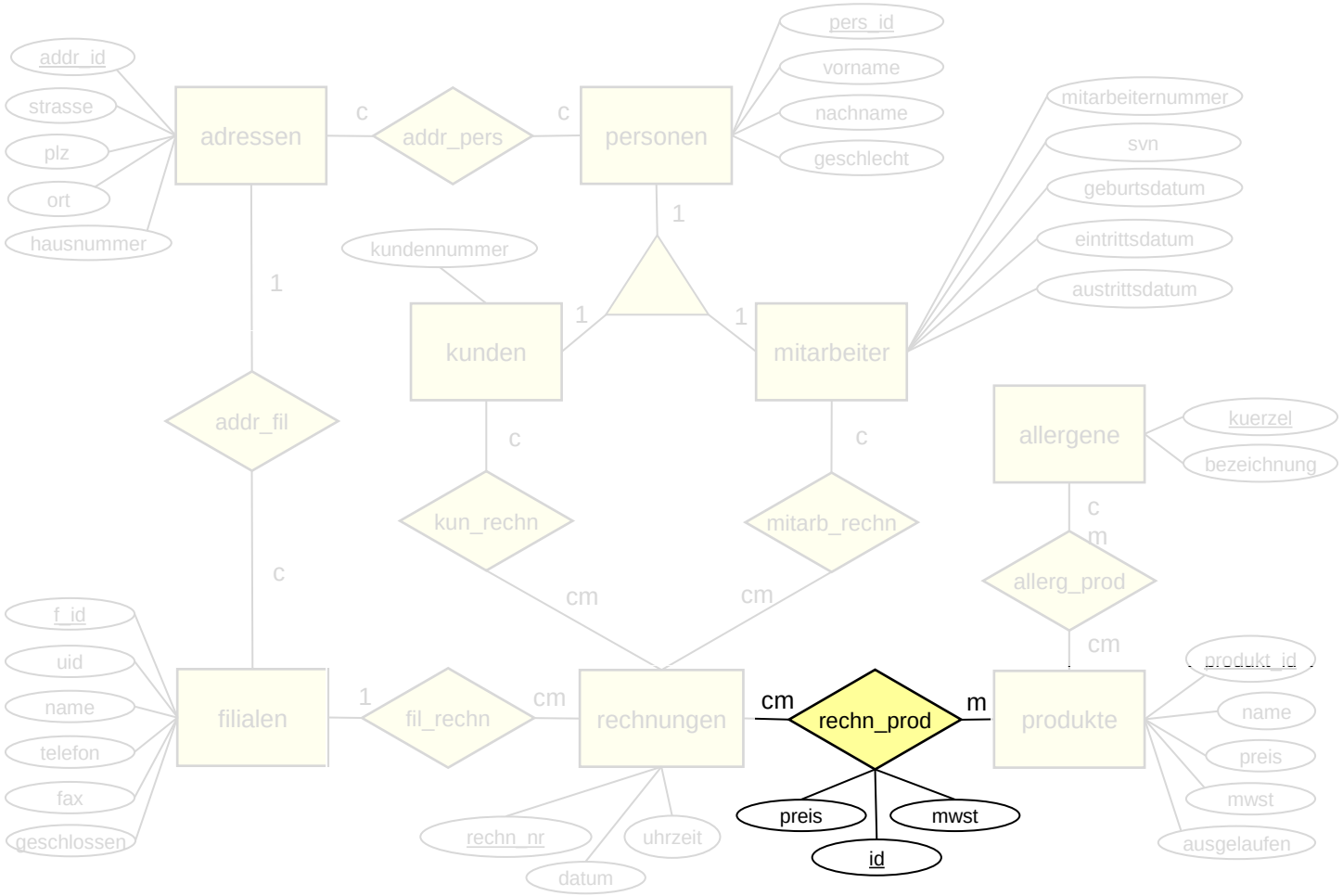
PostgreSQL

```
CREATE TABLE IF NOT EXISTS allergene (
  kuerzel CHAR(1) NOT NULL,
  bezeichnung VARCHAR(64) NOT NULL,
  PRIMARY KEY (kuerzel),
  CONSTRAINT ck_kuerzel
    CHECK (kuerzel IN ('A', 'B', 'C', ,D', 'E',
         'F','G','H','L','M','N','O','P','R')));
```

- **allergene** (<u>kuerzel</u>, bezeichnung)

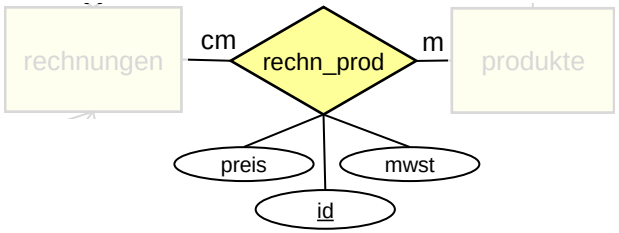


TOPRAST – Tabelle rech_prod



TOPRAST – Tabelle rech_prod

- **rechn_prod** (<u>id</u>, preis, mwst, *#fk_prod_id*, *#fk_rechn_nr*)



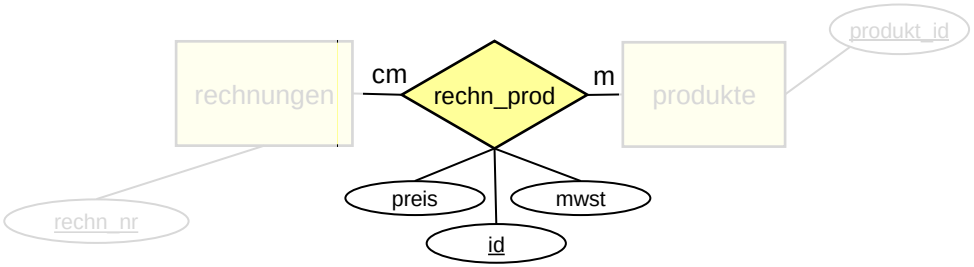
TOPRAST – Create Statement rech_prod

Oracle

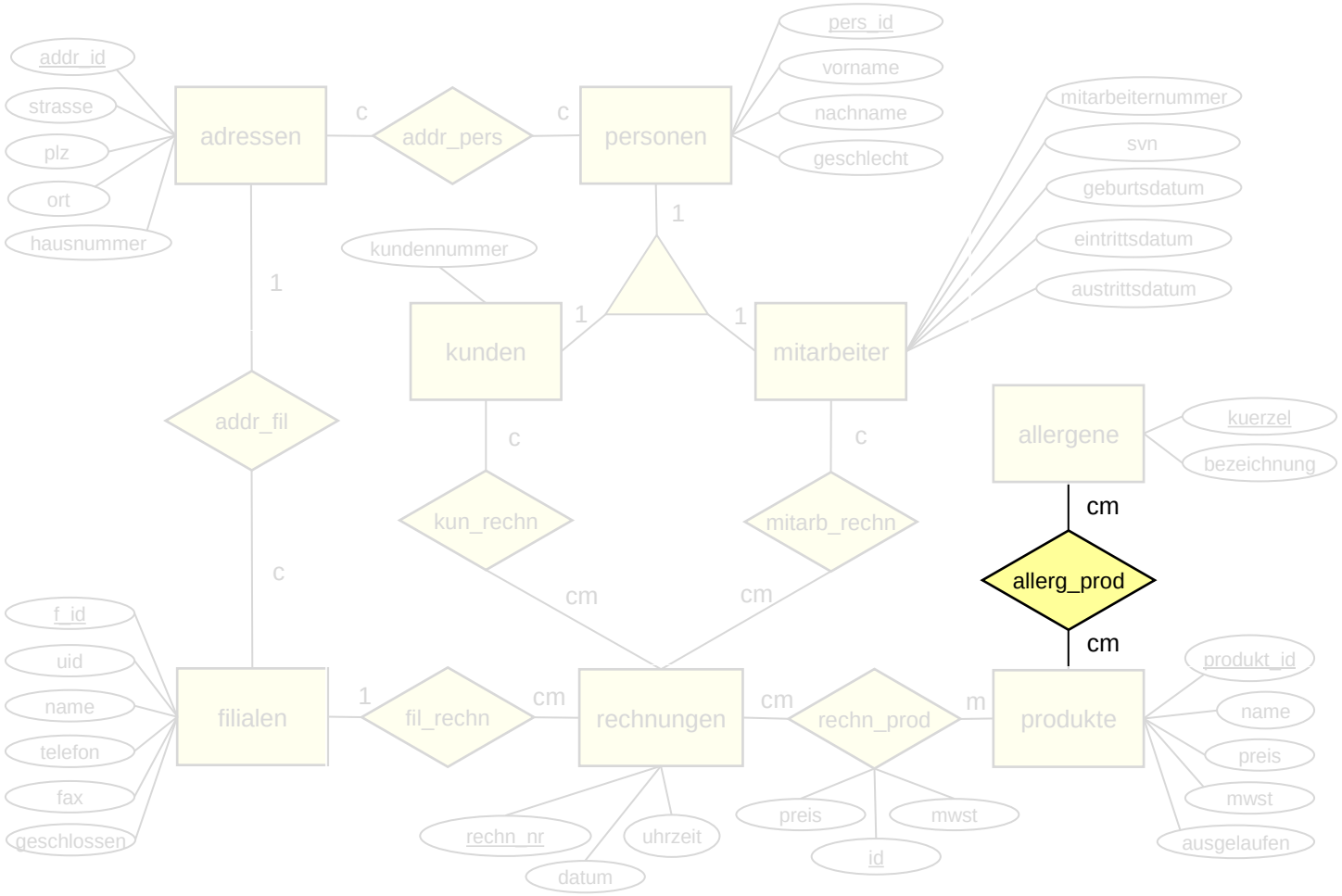
```
CREATE TABLE rechn_prod (
  id INT NOT NULL,
  preis REAL NOT NULL,
  mwst REAL NOT NULL,
  fk_prod_id INT NULL,
  fk_rechn_nr INT NOT NULL,
  PRIMARY KEY (id),
  CONSTRAINT fk_rechn_prod_rechnungen
    FOREIGN KEY (fk_rechn_nr)
    REFERENCES rechnungen (rechn_nr)
    ON DELETE CASCADE,
  CONSTRAINT fk_rechn_prod_produkte
    FOREIGN KEY (fk_prod_id)
    REFERENCES produkte (produkt_id)
    ON DELETE SET NULL);
```

PostgreSQL

```
CREATE TABLE IF NOT EXISTS rechn_prod (
  id SERIAL NOT NULL,
  preis REAL NOT NULL,
  mwst REAL NOT NULL,
  fk_prod_id INT NULL,
  fk_rechn_nr INT NOT NULL,
  PRIMARY KEY (id),
  CONSTRAINT fk_rechn_prod_rechnungen
    FOREIGN KEY (fk_rechn_nr)
    REFERENCES rechnungen (rechn_nr)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT fk_rechn_prod_produkte
    FOREIGN KEY (fk_prod_id)
    REFERENCES produkte (produkt_id)
    ON DELETE SET NULL
    ON UPDATE SET NULL);
```

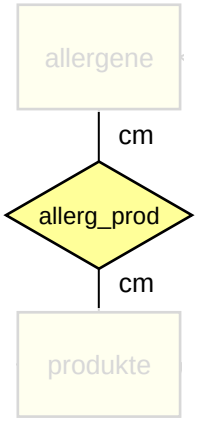


TOPRAST – Tabelle allerg_prod



TOPRAST – Tabelle allerg_prod

- **allerg_prod** (#*fk_kuerzel*, #*fk_prod_id*)



TOPRAST – Create Statement allerg_prod

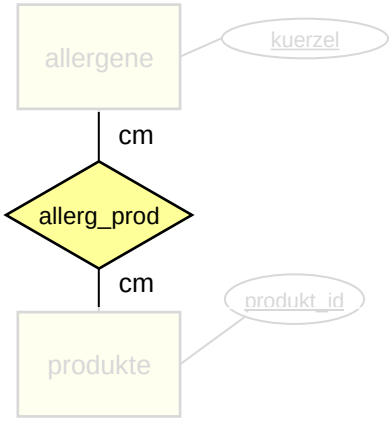
Oracle

```
CREATE TABLE allerg_prod (
  fk_kuerzel CHAR(1) NOT NULL,
  fk_produkt_id INT NOT NULL,
  PRIMARY KEY (fk_kuerzel, fk_produkt_id),
  CONSTRAINT fk_allerg_prod_allergene
    FOREIGN KEY (fk_kuerzel)
    REFERENCES allergene (kuerzel)
    ON DELETE CASCADE,
  CONSTRAINT fk_allerg_prod_produkte
    FOREIGN KEY (fk_produkt_id)
    REFERENCES produkte (produkt_id)
    ON DELETE CASCADE);
```

PostgreSQL

```
CREATE TABLE IF NOT EXISTS allerg_prod (
  fk_kuerzel CHAR(1) NOT NULL,
  fk_produkt_id INT NOT NULL,
  PRIMARY KEY (fk_kuerzel, fk_produkt_id),
  CONSTRAINT fk_allerg_prod_allergene
    FOREIGN KEY (fk_kuerzel)
    REFERENCES allergene (kuerzel)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT fk_allerg_prod_produkte
    FOREIGN KEY (fk_produkt_id)
    REFERENCES produkte (produkt_id)
    ON DELETE CASCADE
    ON UPDATE CASCADE);
```

- **allerg_prod** (*#fk_kuerzel, #fk_prod_id*)



Erstellen von Tabellen

- werden PK – FK Beziehungen gleich beim erstellen von Tabellen definiert, muss auf die Reihenfolge beim Erstellen der Tabellen geachtet werden.
  - Das Erstellen einer Tabelle mit FK Referenz auf eine noch nicht erstellte Tabelle führt zu einer Fehlermeldung (Constraint violation)

- Lösung
  - zuerst die Tabellen ohne PK – FK Beziehungen erstellen
  - PK – FK Beziehungen danach mit ALTER TABLE Statements hinzufügen

Create Table – Alter Table

```
-- -----------------------------------------------------
-- Table orte
-- -----------------------------------------------------

CREATE TABLE orte (
  plz INT NOT NULL,
  ort VARCHAR(64) NOT NULL,
  PRIMARY KEY (plz),
  CONSTRAINT ck_plz
          CHECK (plz between 1000 and 9999));


-- -----------------------------------------------------
-- Table adressen
-- -----------------------------------------------------

CREATE TABLE adressen (
  addr_id INT NOT NULL,
  strasse VARCHAR(64) NOT NULL,
  hausnummer VARCHAR(10) NULL,
  plz int NOT NULL,
  PRIMARY KEY (addr_id));
```

Create Table – Alter Table

```
-- -----------------------------------------------------
-- Table personen
-- -----------------------------------------------------

CREATE TABLE personen (
  pers_id INT PRIMARY KEY NOT NULL,
  vorname VARCHAR(45) NULL,
  nachname VARCHAR(45) NULL,
  geschlecht CHAR(1)
    CONSTRAINT ck_personen_geschlecht
      CHECK (geschlecht IN('m','w')),
  fk_addr_id INT);


-- -----------------------------------------------------
-- Table kunden
-- -----------------------------------------------------

CREATE TABLE kunden (
  fk_pers_id INT NOT NULL,
  kundennummer INT UNIQUE NOT NULL,
  PRIMARY KEY (fk_pers_id));
```

Create Table – Alter Table

```
-- ------------------------------------------------------
-- Table mitarbeiter
-- ------------------------------------------------------

CREATE TABLE mitarbeiter (
  fk_pers_id INT NOT NULL,
  mitarbeiternummer INT UNIQUE NOT NULL,
  svn INT NOT NULL,
  geburtsdatum DATE NOT NULL,
  eintrittsdatum DATE NOT NULL,
  austrittsdatum DATE NULL,
  PRIMARY KEY (fk_pers_id),
  CONSTRAINT ck_mitarbeiter_svn
    CHECK (svn BETWEEN 1000 AND 9999));
```

Create Table – Alter Table

```
-- -----------------------------------------------------
-- Table filialen
-- -----------------------------------------------------

CREATE TABLE filialen (
  f_id INT NOT NULL,
  "uid" VARCHAR(12) NULL,
  name VARCHAR(45) NOT NULL,
  telefon VARCHAR(45) NULL,
  fax VARCHAR(45) NULL,
  fk_addr_id INT NOT NULL,
  geschlossen CHAR(1) DEFAULT 0 NOT NULL,
  PRIMARY KEY (f_id),
  CONSTRAINT ck_filialen_geschlossen
    CHECK (CAST(geschlossen AS INT) IN (0,1)));
```

Create Table – Alter Table

```
-- ------------------------------------------------------
-- Table rechnungen
-- ------------------------------------------------------

CREATE TABLE rechnungen (
  rechn_nr INT NOT NULL,
  datum DATE NOT NULL,
  zeit TIMESTAMP NOT NULL,
  fk_f_id INT NOT NULL,
  fk_kunden_pers_id INT NULL,
  fk_mitarbeiter_pers_id INT NULL,
  PRIMARY KEY (rechn_nr));

-- ------------------------------------------------------
-- Table produkte
-- ------------------------------------------------------

CREATE TABLE produkte (
  produkt_id INT NOT NULL,
  name VARCHAR(64) NOT NULL,
  preis REAL NOT NULL,
  mwst REAL NOT NULL,
  ausgelaufen CHAR(1) DEFAULT 0 NOT NULL,
  PRIMARY KEY (produkt_id),
  CONSTRAINT ck_produkte_ausgelaufen
    CHECK (CAST(ausgelaufen AS INT) IN (0,1)));
```

Create Table – Alter Table

```
-- -----------------------------------------------------
-- Table rechn_prod
-- -----------------------------------------------------

CREATE TABLE rechn_prod (
  id INT NOT NULL,
  preis REAL NOT NULL,
  mwst REAL NOT NULL,
  fk_prod_id INT NULL,
  fk_rechn_nr INT NOT NULL,
  PRIMARY KEY (id));


-- -----------------------------------------------------
-- Table allergene
-- -----------------------------------------------------

CREATE TABLE allergene (
  kuerzel CHAR(1) NOT NULL,
  bezeichnung VARCHAR(64) NOT NULL,
  PRIMARY KEY (kuerzel),
  CONSTRAINT ck_allergene_kuerzel
                CHECK (kuerzel in ('A','B','C','D','E','F','G','H','L','M','N','O','P','R')));


-- -----------------------------------------------------
-- Table allerg_prod
-- -----------------------------------------------------

CREATE TABLE allerg_prod (
  fk_kuerzel CHAR(1) NOT NULL,
  fk_produkt_id INT NOT NULL,
  PRIMARY KEY (fk_kuerzel, fk_produkt_id));
```

Create Table – Alter Table

```
-- ------------------------------------------------------
-- Alter Table adressen
-- ------------------------------------------------------

ALTER TABLE adressen
  ADD CONSTRAINT fk_plz
    FOREIGN KEY (plz)
      REFERENCES orte (plz);


-- ------------------------------------------------------
-- Alter Table personen
-- ------------------------------------------------------

ALTER TABLE personen
  ADD CONSTRAINT fk_personen_adressen
    FOREIGN KEY (fk_addr_id)
      REFERENCES adressen (addr_id)
      ON DELETE SET NULL;
```

Create Table – Alter Table

```
-- ------------------------------------------------------
-- Alter Table kunden
-- ------------------------------------------------------

ALTER TABLE kunden
  ADD CONSTRAINT fk_kunden_personen
    FOREIGN KEY (fk_pers_id)
      REFERENCES personen (pers_id)
      ON DELETE CASCADE;


-- ------------------------------------------------------
-- Alter Table filialen
-- ------------------------------------------------------

ALTER TABLE filialen
  ADD CONSTRAINT fk_filialen_adressen
        FOREIGN KEY (fk_addr_id)
          REFERENCES adressen (addr_id);
```

Create Table – Alter Table

```
-- ------------------------------------------------------
-- Alter Table mitarbeiter
-- ------------------------------------------------------

ALTER TABLE mitarbeiter
  ADD CONSTRAINT fk_mitarbeiter_personen
        FOREIGN KEY (fk_pers_id)
          REFERENCES personen (pers_id);


-- ------------------------------------------------------
-- Alter Table rechnungen
-- ------------------------------------------------------

ALTER TABLE rechnungen
  ADD CONSTRAINT fk_rechnungen_filialen
        FOREIGN KEY (fk_f_id)
          REFERENCES filialen (f_id)
  ADD CONSTRAINT fk_rechnungen_kunden
        FOREIGN KEY (fk_kunden_pers_id)
          REFERENCES kunden (fk_pers_id)
            ON DELETE SET NULL
  ADD CONSTRAINT fk_rechnungen_mitarbeiter
      FOREIGN KEY (fk_mitarbeiter_pers_id)
        REFERENCES mitarbeiter (fk_pers_id)
          ON DELETE SET NULL;
```

Create Table – Alter Table

```
-- ------------------------------------------------
-- Alter Table rechn_prod
-- ------------------------------------------------

ALTER TABLE rechn_prod
  ADD CONSTRAINT fk_rechn_prod_rechnungen
        FOREIGN KEY (fk_rechn_nr)
          REFERENCES rechnungen (rechn_nr)
            ON DELETE CASCADE
  ADD CONSTRAINT fk_rechn_prod_produkte
        FOREIGN KEY (fk_prod_id)
          REFERENCES produkte (produkt_id)
            ON DELETE SET NULL;


-- ------------------------------------------------
-- Alter Table allerg_prod
-- ------------------------------------------------

ALTER TABLE allerg_prod
  ADD CONSTRAINT fk_allerg_prod_allergene
        FOREIGN KEY (fk_kuerzel)
          REFERENCES allergene (kuerzel)
            ON DELETE CASCADE
  ADD CONSTRAINT fk_allerg_prod_produkte
        FOREIGN KEY (fk_produkt_id)
          REFERENCES produkte (produkt_id)
            ON DELETE CASCADE;
```

Daten einfügen (INSERT)

- Beim Einfügen der Daten muss wieder auf die Reihenfolge geachtet werden, damit es nicht zu Verletzungen der Referentiellen Integrität (PK – FK Beziehungen) kommt

- Daten können einzeln eingefügt werden
  - ein Datensatz je INSERT Statement

- es können auch mehrere Datensätze mit einem Statement eingefügt werden
  - unterschiedliche Behandlung in Oracle und PostgreSQL