## *Wiederholung: Tabellendefinition in SQL*

### *Tabelle löschen*

```
DROP TABLE tabellenname;
```

### *Tabelle ändern*

Ermöglicht das Hinzufügen (`ADD`), Ändern (`MODIFY`) oder Löschen (`DROP`) von Spalten

```
ALTER TABLE tabellenname
   ADD (attribut datentyp); |
   MODIFY (attribut datentyp); |
   DROP (attribut);
```

### *Tabelle anlegen (vereinfacht)*

```
CREATE TABLE tabellenname (
   attribut_1 datentyp_1 [constraint_11] [,...],
   attribut_2 datentyp_2 [constraint_21] [,...],
        ...,
   attribut_k datentyp_k [constraint_k1] [,...],
   [tabellenconstraint_1, ...,tabellenconstraint_m]
   );
```

- erzeugt eine neue leere Tabelle
- *attribut*$_i$: Name des $i$-ten Attributs
- *datentyp*$_i$: Datentyp des $i$-ten Attributs

  Wichtige Datentypen in SQL:

| | |
|---|---|
| `CHAR(`$n$`)` | String der festen Länge $n$ |
| `VARCHAR(`$n$`)` | variabler String mit max. Länge $n$ |
| `INT` | ganze Zahl |
| `DEC(`$n$`,`$m$`)` | Festkommazahl mit insges. $n$ Stellen, davon $m$ Nachkommastellen |
| `FLOAT(`$m$`)` | Gleitkommazahl mit max. $m$ Nachkommastellen |
| `DATE` | Datum |
| `TIME` | Zeit |

- *constraint*$_{ij}$: optionaler Constraint für das $i$-te Attribut

| | |
|---|---|
| `NOT NULL` | Attribut muss gefüllt werden |
| `UNIQUE` | Attributwert darf in der Tabelle nicht doppelt vorkommen |
| `PRIMARY KEY` | Attribut ist Primärschlüssel |
| `CHECK(`$b$`)` | Attributwert muss Bedingung $b$ erfüllen |
| `DEFAULT=`$x$ | setzt Defaultwert $x$ für das Attribut |
| `REFERENCES` $t$`(`$a$`)` | markiert das Attribut als Fremdschlüssel, der auf das Attribut $a$ der Tabelle $t$ verweist ($a$ muss unique oder Primärschlüssel sein) |

- *tabellenconstraint*$_i$: optionaler Constraint für die Tabelle (bezieht sich i.d.R. auf mehrere Attribute)

| | |
|---|---|
| `PRIMARY KEY (`$a_1,\ldots,a_k$`)` | Attribute $a_1,\ldots,a_k$ bilden zusammengesetzten Primärschlüssel |
| `FOREIGN KEY (`$a_1,\ldots,a_k$`)` `REFERENCES` $t$`(`$b_1,\ldots,b_k$`)` | markiert die Attribute $a_1,\ldots,a_k$ als Fremdschlüssel, die auf die Attribute $b_1,\ldots,b_k$ der Tabelle $t$ verweisen ($b_1,\ldots,b_k$ müssen unique oder Primärschlüssel sein) |
| `UNIQUE (`$a_1,\ldots,a_k$`)` | Attributwerte $a_1,\ldots,a_k$ dürfen nicht doppelt vorkommen |
| `CHECK(`$b$`)` | Bedingung $b$ muss erfüllt sein |

## *Wiederholung: Relationale Algebra*

- Formale Sprache für Anfragen über einem relationalem Schema
- Definition von Operationen auf einer Menge von Relationen, die Ergebnisse aller Operationen sind ebenfalls Relationen ($\rightarrow$ abgeschlossen)
- praktische Umsetzung: Sprache SQL (Structured Query Language)
- Grundoperationen
  - Vereinigung: $R \cup S = \{t | t \in R \text{ oder } t \in S\}$
  - Differenz: $R - S = \{t | t \in R \text{ und } t \notin S\}$
  - Kartesisches Produkt:
    $R \times S = \{(a_1, \ldots, a_r, a_{r+1}, \ldots, a_{r+s}) | (a_1, \ldots, a_r) \in R \text{ und } (a_{r+1}, \ldots, a_{r+s}) \in S\}$
  - Selektion: $\sigma_F(R) = \{t | t \in R \text{ und } t \text{ erfüllt Formel } F\}$
    ($F$ besteht aus Konstanten, Attributen, Vergleichsoperatoren und Booleschen Operatoren)
  - Projektion: $\Pi_{a_1,\ldots,a_m}(R) = \{t[a_1, \ldots a_m] | t \in R\}$,
    wobei $t[a_1, \ldots a_m]$ ein Tupel aus $R$ bezeichnet, das nur die Attributwerte $a_1, \ldots a_m$ enthält
- Weitere Operationen
  - Durchschnitt: $R \cap S = \{t | t \in R \text{ und } t \in S\}$
  - Quotient: $R \div S = \{t | t \in \Pi_{R-S}(R) \text{ und } \{t\} \times S \subseteq R\}$
    Bildlich: Ergebnis enthält alle linken Hälften von Tupel aus $R$, die mit allen rechten Hälften von $S$ kombiniert in $R$ auftreten
  - Theta-Join: $R \underset{A\theta B}{\bowtie} S = \sigma_{A\theta B}(R \times S)$ ($\theta \in \{=, <, \leq, >, \geq, \neq\}$
  - Equi-Join: $R \underset{A=B}{\bowtie} S$
  - Natural-Join $R \bowtie S$: Equi-Join bzgl. aller gleichnamigen Attribute in $R$ und $S$, gleiche Spalten werden entfernt