

20 SQL-Cheat-Sheet

20.1 Einfache SQL-Abfragen

```
SELECT username AS "Name", city AS "Stadt"
FROM users
WHERE city = "Berlin" AND NOT name LIKE "%Ma?er"
```

- Logische Operatoren: AND, OR, NOT
- Vergleiche: >, <, >=, <=, =, LIKE
- Duplikate in einer Spalte ausschließen: DISTINCT
- Sortieren: ORDER BY
- Zahl der Ausgabezeilen begrenzen: LIMIT

20.2 Aggregate

```
SELECT city, MIN(centimeters), MAX(centimeters)
FROM users
GROUP BY city
```

- Aggregatfunktionen: AVG, MIN, MAX, COUNT, SUM
- Aggregate filtern: HAVING

20.3 CRUD

20.3.1 INSERT

```
INSERT
INTO users (
    username, email, password,
    name, bio, gender,
    birthday, city, country,
    centimeters, avatar, role,
    is_active, remember_token, created_at,
    updated_at
)
VALUES (
    'guenther37', 'guenther@instahub.app', '12345',
    'Günther Müller', 'Günther mag Kartoffelsalat.',
    'male',
    '2006-06-06 00:00:00', 'Leipzig', 'Deutschland',
    '173', 'avatar.png', 'user',
    '0', NULL, now(),

```

```
now()  
)
```

20.3.2 UPDATE

```
UPDATE users  
SET country = "Deutschland"  
WHERE country = "DDR"
```

20.3.3 DELETE

```
DELETE FROM users  
WHERE birthday > "2018-01-01"
```

20.3.4 CREATE

```
CREATE TABLE photos (  
    id INT(4) UNSIGNED NOT NULL AUTO_INCREMENT,  
    user_id INT(4) UNSIGNED NOT NULL,  
    description VARCHAR(255) NOT NULL,  
    url VARCHAR(255) NOT NULL,  
    created_at TIMESTAMP NOT NULL DEFAULT now(),  
    updated_at TIMESTAMP NOT NULL DEFAULT now(),  
    PRIMARY KEY (id),  
    FOREIGN KEY (user_id)  
        REFERENCES users(id)  
        ON DELETE CASCADE  
)
```

- Datentypen:
 - INT - ganze Zahlen
 - SMALL INT - ganze Zahlen, meist zwischen -32.768 und +32.767
 - TIMESTAMP - Datum und Uhrzeit
 - VARCHAR(*n*) - Zeichenkette der variablen Länge *n*
 - DECIMAL(*p*, *s*) - *p* Stellen insgesamt, *s* davon sind Nachkommastellen
 - FLOAT - Gleitkommazahl (wie beim Taschenrechner)
 - CHAR (*n*) - Zeichenkette fester Länge

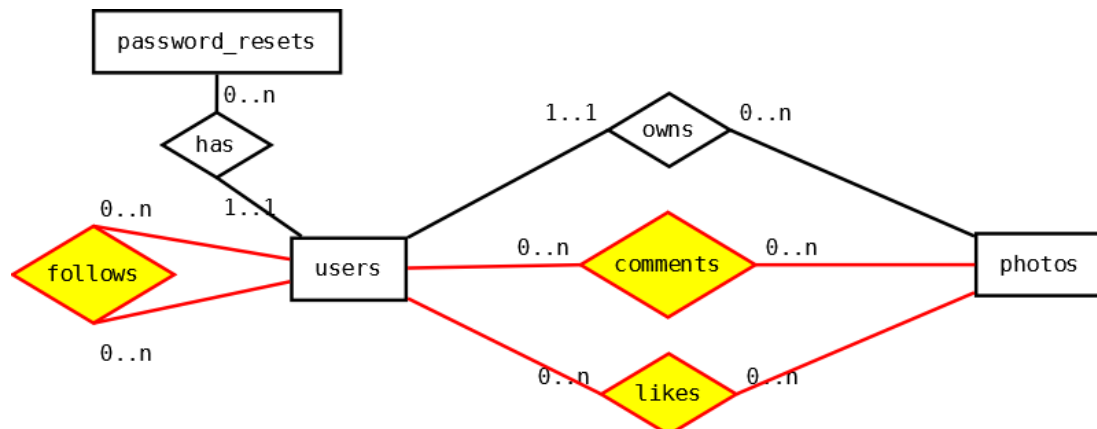
20.4 JOIN

```
SELECT count(users.id)  
FROM users JOIN photos  
ON (users.id = user_id)
```

```
WHERE username="aaron113"
```

- LEFT JOIN - Alle Datensätze der linken Tabelle, auch wenn es keinen passenden Datensatz in der rechten Tabelle gibt.
- RIGHT JOIN
- INNER JOIN - identisch mit JOIN
- NATURAL JOIN - identisch mit JOIN, wobei alle Spalten mit identischen Namen in die ON-Klausel übernommen werden.

20.5 ER-Diagramme



Beispiel für ein ER-Diagramm