SQL-Abfragen – Kurz-Cheat-Sheet (Code-Schemata)

1. SELECT / FROM / WHERE

1.1 Alle Spalten oder bestimmte Spalten

```
-- alle Spalten
SELECT *
FROM tabelle;

-- bestimmte Spalten
SELECT spalte1, spalte2, spalte3
FROM tabelle;
```

1.2 Auswahl nach Bedingung (WHERE)

```
SELECT spalte1, spalte2
FROM tabelle
WHERE spalte_x operator wert;
-- operator: =  <>  <  <=  >  >=
```

2. Rechenoperationen in SELECT

```
-- einfache Rechnung
SELECT zahl1 + zahl2;

-- Rechnung mit Spalten
SELECT spalte_x + 5         AS neu_x,
       spalte_y * 1.2       AS faktor_y
FROM tabelle;

-- ganzzahlige Division
SELECT zahl1 DIV zahl2;     -- Quotient (ganzzahlig)
SELECT zahl1 MOD zahl2;     -- Rest
```

3. BETWEEN / IN / LIKE

3.1 BETWEEN (inklusive Grenzen)

```
SELECT spalte1, spalte2
FROM tabelle
WHERE spalte_z BETWEEN untergrenze AND obergrenze;
-- funktioniert mit Zahlen, Datumswerten und Zeichenketten
```

3.2 IN (Wert aus Liste)

```
SELECT spalte1, spalte2
FROM tabelle
WHERE spalte_x IN (wert1, wert2, wert3);

```

```sql
SELECT spalte1
FROM tabelle
WHERE textspalte IN ('EintragA', 'EintragB', 'EintragC');
```

3.3 LIKE (Mustervergleich)

```sql
-- %  : beliebig viele Zeichen
-- _  : genau ein Zeichen

-- beginnt mit 'A'
SELECT *
FROM tabelle
WHERE textspalte LIKE 'A%';

-- endet mit 'X'
SELECT *
FROM tabelle
WHERE textspalte LIKE '%X';

-- enth lt 'mitte'
SELECT *
FROM tabelle
WHERE textspalte LIKE '%mitte%';

-- genau 5 Zeichen, beginnt mit 'B'
SELECT *
FROM tabelle
WHERE textspalte LIKE 'B____';
```

4. AND / OR / NOT

```sql
-- UND-Verknuepfung (AND)
SELECT *
FROM tabelle
WHERE bedingung1
  AND bedingung2;

-- ODER-Verknuepfung (OR)
SELECT *
FROM tabelle
WHERE bedingung1
   OR bedingung2;

-- Negation (NOT)
SELECT *
FROM tabelle
WHERE NOT bedingung;

-- Bereich mit AND (Alternative zu BETWEEN)
SELECT *
FROM tabelle
WHERE spalte BETWEEN 10 AND 100;

SELECT *
FROM tabelle
WHERE spalte >= 10
  AND spalte <= 100;
```

5. ORDER BY / LIMIT

5.1 ORDER BY

```sql
-- aufsteigend sortieren (Standard)
SELECT *
FROM tabelle
ORDER BY spalte1;

-- absteigend sortieren
SELECT *
FROM tabelle
ORDER BY spalte1 DESC;

-- nach mehreren Spalten sortieren
SELECT *
FROM tabelle
ORDER BY spalte1 ASC,
         spalte2 DESC;
```

5.2 LIMIT (Ergebnismenge beschränken)

```sql
-- erste n Zeilen
SELECT *
FROM tabelle
ORDER BY spalte1
LIMIT n;

-- mit Offset: ueberspringe offset Zeilen, dann n Zeilen ausgeben
SELECT *
FROM tabelle
ORDER BY spalte1
LIMIT offset, n;
```

6. Aggregatfunktionen und GROUP BY

6.1 Aggregatfunktionen

```sql
SELECT COUNT(*)          AS anzahl_zeilen,
       SUM(spalte_x)     AS summe,
       AVG(spalte_x)     AS mittelwert,
       MIN(spalte_x)     AS minimum,
       MAX(spalte_x)     AS maximum
FROM tabelle;
```

6.2 GROUP BY

```sql
-- Aggregation pro Gruppe (z.B. pro kategorie)
SELECT gruppen_spalte,
       COUNT(*)      AS anzahl,
       AVG(wert_spalte) AS durchschnitt
FROM tabelle
GROUP BY gruppen_spalte;

-- Aggregation + Sortierung
SELECT gruppen_spalte,
       SUM(wert_spalte) AS summe
FROM tabelle
GROUP BY gruppen_spalte
ORDER BY summe DESC;
```

7. Joins (Tabellen verknuepfen)

7.1 CROSS JOIN (kartesisches Produkt)

```sql
SELECT *
FROM tabelle_a
CROSS JOIN tabelle_b;
```

7.2 INNER JOIN

```sql
-- Variante mit JOIN
SELECT a.spalte1, b.spalte2
FROM tabelle_a AS a
INNER JOIN tabelle_b AS b
    ON a.fk_spalte = b.pk_spalte;

-- Variante mit WHERE (logisch gleich)
SELECT a.spalte1, b.spalte2
FROM tabelle_a AS a, tabelle_b AS b
WHERE a.fk_spalte = b.pk_spalte;
```

7.3 LEFT / RIGHT OUTER JOIN

```sql
-- LEFT OUTER JOIN:
-- alle Zeilen aus linker Tabelle (tabelle_a),
-- passende Zeilen aus rechter Tabelle (tabelle_b),
-- sonst NULL in den Spalten von b
SELECT a.spalte1, b.spalte2
FROM tabelle_a AS a
LEFT OUTER JOIN tabelle_b AS b
    ON a.fk_spalte = b.pk_spalte;

-- RIGHT OUTER JOIN:
-- alle Zeilen aus rechter Tabelle (tabelle_b),
-- passende Zeilen aus linker Tabelle (tabelle_a),
-- sonst NULL in den Spalten von a
SELECT a.spalte1, b.spalte2
FROM tabelle_a AS a
RIGHT OUTER JOIN tabelle_b AS b
    ON a.fk_spalte = b.pk_spalte;
```