Kapitel 5 – SQL-Auswertung

5.1 Aggregatfunktionen

Definition

Aggregatfunktionen berechnen einen einzelnen Ergebniswert aus mehreren Datensätzen.

Sie werden häufig zusammen mit **GROUP BY** verwendet.

Wichtige Aggregatfunktionen

Funktion	Bedeutung
COUNT()	zählt Datensätze
SUM()	berechnet die Summe
AVG()	berechnet den Durchschnitt
MIN()	ermittelt den kleinsten Wert
MAX()	ermittelt den größten Wert

COUNT

```
SELECT COUNT(*)
FROM kunde;
```

Zählt alle Datensätze der Tabelle **kunde**.

SUM

```
SELECT SUM(preis)
FROM artikel;
```

Berechnet die Summe aller Werte in der Spalte **preis**.

AVG

```
SELECT AVG(preis)
FROM artikel;
```

Berechnet den Durchschnitt der Werte in der Spalte **preis**.

MIN und MAX

```
SELECT MIN(preis), MAX(preis)
FROM artikel;
```

Ermittelt den kleinsten und größten Preis.

Merksatz

Aggregatfunktionen fassen mehrere Datensätze zu einem Ergebnis zusammen.

Prüfungshinweis

In IHK-Aufgaben wird häufig gefragt, welche Funktion für Zählen, Summe oder Durchschnitt verwendet wird.

Typische Fehler:

- COUNT und SUM verwechseln.
- AVG für die Summe verwenden.
- Aggregatfunktionen ohne GROUP BY falsch interpretieren.
- WHERE und HAVING verwechseln.

Quellen

- IT-Handbuch für Fachinformatiker

5.2 GROUP BY

Definition

GROUP BY fasst Datensätze mit gleichen Werten zu Gruppen zusammen.

Es wird häufig zusammen mit **Aggregatfunktionen** verwendet.

Grundaufbau

```
SELECT spalte, aggregatfunktion
FROM tabelle
GROUP BY spalte;
```

Beispiel

```
SELECT ort, COUNT(*)
FROM kunde
GROUP BY ort;
```

Bedeutung:

Die Kunden werden nach **Ort** gruppiert.
Für jeden Ort wird die Anzahl der Kunden gezählt.

Typische Verwendung

GROUP BY wird verwendet für:

- Anzahl pro Gruppe
- Summe pro Gruppe
- Durchschnitt pro Gruppe
- kleinster oder größter Wert pro Gruppe

Beispiel mit SUM

```
SELECT kundennr, SUM(betrag)
FROM rechnung
```

```
GROUP BY kundennr;
```

Bedeutung:

Für jeden Kunden wird die Summe seiner Rechnungsbeträge berechnet.

Merksatz

GROUP BY bildet Gruppen aus Datensätzen.

Prüfungshinweis

In IHK-Aufgaben wird **GROUP BY** häufig zusammen mit **COUNT**, **SUM** oder **AVG** verwendet.

Typische Fehler:

- GROUP BY vergessen.
- GROUP BY mit ORDER BY verwechseln.
- WHERE und HAVING verwechseln.
- Spalten im SELECT verwenden, die nicht gruppiert oder aggregiert sind.

Quellen

- IT-Handbuch für Fachinformatiker

5.3 HAVING

Definition

HAVING filtert gruppierte Ergebnisse nach einer Bedingung.

HAVING wird zusammen mit **GROUP BY** und Aggregatfunktionen verwendet.

Grundaufbau

```
SELECT spalte, aggregatfunktion
FROM tabelle
GROUP BY spalte
HAVING bedingung;
```

Beispiel

```
SELECT ort, COUNT(*)
FROM kunde
GROUP BY ort
HAVING COUNT(*) > 5;
```

Bedeutung:

Es werden nur Orte angezeigt, in denen mehr als **5 Kunden** vorhanden sind.

WHERE und HAVING

Klausel	Aufgabe
WHERE	filtert einzelne Datensätze vor der Gruppierung
HAVING	filtert Gruppen nach der Gruppierung

Beispiel WHERE

```
SELECT *
FROM kunde
```

```
WHERE ort = 'Berlin';
```

Filtert einzelne Kunden aus Berlin.

Beispiel HAVING

```
SELECT ort, COUNT(*)
FROM kunde
GROUP BY ort
HAVING COUNT(*) > 5;
```

Filtert Gruppen mit mehr als 5 Kunden.

Merksatz

WHERE filtert Datensätze.
HAVING filtert Gruppen.

Prüfungshinweis

In IHK-Aufgaben wird häufig der Unterschied zwischen **WHERE** und **HAVING** abgefragt.

Typische Fehler:

- HAVING ohne GROUP BY verwenden.
- Aggregatfunktionen in WHERE verwenden.
- WHERE und HAVING verwechseln.
- HAVING vor GROUP BY schreiben.

Quellen

- IT-Handbuch für Fachinformatiker

5.4 SQL-Reihenfolge

Definition

Die **SQL-Reihenfolge** beschreibt, in welcher Reihenfolge SQL-Klauseln in einer Abfrage geschrieben werden.

Schreibreihenfolge

```
SELECT spalte
FROM tabelle
WHERE bedingung
GROUP BY spalte
HAVING gruppenbedingung
ORDER BY spalte;
```

Bedeutung

Klausel	Aufgabe
SELECT	legt fest, welche Spalten ausgegeben werden
FROM	legt fest, aus welcher Tabelle gelesen wird
WHERE	filtert einzelne Datensätze
GROUP BY	gruppiert Datensätze
HAVING	filtert Gruppen
ORDER BY	sortiert das Ergebnis

Wichtige Reihenfolge

1. SELECT
2. FROM
3. WHERE
4. GROUP BY
5. HAVING
6. ORDER BY

Beispiel

```
SELECT ort, COUNT(*)
FROM kunde
WHERE aktiv = 1
GROUP BY ort
HAVING COUNT(*) > 5
ORDER BY ort;
```

Merksatz

WHERE kommt vor GROUP BY.
HAVING kommt nach GROUP BY.
ORDER BY steht am Ende.

Prüfungshinweis

In IHK-Aufgaben wird häufig geprüft, ob SQL-Klauseln in der richtigen Reihenfolge verwendet werden.

Typische Fehler:

- WHERE nach GROUP BY schreiben.
- HAVING vor GROUP BY schreiben.
- ORDER BY vor WHERE schreiben.
- Aggregatfunktionen in WHERE verwenden.
- GROUP BY und ORDER BY verwechseln.

Quellen

- IT-Handbuch für Fachinformatiker