SQL Befehle

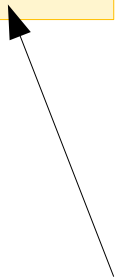
Photo by Tobias Fischer on Unsplash

SQL-Abfragen

Datenbank vorab auswählen sonst muss in SQL-Abfragen `dbname.tabellenname` verwendet werden. (Warum?)

Allgemeine Form eines SQL-Select-Statements:

```
SELECT [DISTINCT] { spalten | * }
FROM              tabelle [alias] [,tabelle [alias]] ...
[WHERE            {bedingung}]
[GROUP BY         spalten [HAVING {bedingung}]]
[ORDER BY         spalten [ASC | DESC]];
```



Redundanzfreie Abfrage (Keine Dopplungen)

SQL-Abfragen: Redundanz entfernen mit DISTINCT

SELECT

```
SELECT Kundenname FROM `Kunden` WHERE KundenName = "Maier"
```

Alles anzeigen | Anzahl der Datensätze: 25

+ Optionen

	Kundenname
Bearbeiten Kopieren Löschen	Maier
Bearbeiten Kopieren Löschen	Maier
Bearbeiten Kopieren Löschen	Maier
Bearbeiten Kopieren Löschen	Maier
Bearbeiten Kopieren Löschen	Maier

SELECT DISTINCT

```
SELECT DISTINCT Kundenname FROM `Kunden` WHERE KundenName = "Maier"
```

Alles anzeigen | Anzahl der Datensätze: 25 Z

+ Optionen

	Kundenname
Bearbeiten Kopieren Löschen	Maier

```
SELECT DISTINCT Kundenname, KundenID FROM `Kunden` WHERE KundenName = "Maier"
```

Alles anzeigen | Anzahl der Datensätze: 25 Zeilen filtern: D

+ Optionen

	Kundenname	KundenID
Bearbeiten Kopieren Löschen	Maier	1
Bearbeiten Kopieren Löschen	Maier	12
Bearbeiten Kopieren Löschen	Maier	13
Bearbeiten Kopieren Löschen	Maier	14
Bearbeiten Kopieren Löschen	Maier	15

Gruppierungen

- Realisierung von Anfragen wie „Ausgabe der Gesamtbestände aller Artikel einer Preiskategorie“

Wieviele Bücher habe ich am Lager, die 1 EUR kosten?

- Man muss alle Bücher wählen deren Preis 1EUR ist
- Die Bestände all dieser Bücher müssen addiert werden

- Reduktion auf einen Repräsentanten mittels der `GROUP BY`-Klausel

Ohne Gruppierung

```
SELECT * FROM artikel ORDER BY APreis ASC
```

←T→	ANr	AName	APreis	ABild	ABestand
Bearbeiten Kopieren Löschen	100009	Visual C# 2010 Shortcuts - Befehlskarte	1.00	93970143A.jpg	100
Bearbeiten Kopieren Löschen	100011	TYPO3 Referenz: Extbase & Fluid - Befehlskarte	1.00	00000233A.jpg	100
Bearbeiten Kopieren Löschen	100010	Visual Basic 2010 Shortcuts - Befehlskarte	1.00	93970145A.jpg	100
Bearbeiten Kopieren Löschen	100008	CSS Referenz - Befehlskarte	1.00	93970156A.jpg	100
Bearbeiten Kopieren Löschen	100007	HTML Referenz - Befehlskarte	1.00	93970155A.jpg	100
Bearbeiten Kopieren Löschen	100006	XML & DTD Referenz - Befehlskarte	1.00	93970135A.jpg	100
Bearbeiten Kopieren Löschen	100005	C# .NET Referenz - Befehlskarte	1.00	93970130A.jpg	100
Bearbeiten Kopieren Löschen	100004	Java Referenz - Befehlskarte	1.00	93970138A.jpg	100
Bearbeiten Kopieren Löschen	100003	XML Schema Referenz - Befehlskarte	1.00	93970128A.jpg	70
Bearbeiten Kopieren Löschen	100002	PHP Referenz - Befehlskarte	1.00	93970129A.jpg	0
Bearbeiten Kopieren Löschen	100012	VBA mit Excel - easy	2.99	00000233A.jpg	50
Bearbeiten Kopieren Löschen	100016	Programmieren lernen für Kinder	9.99	82724415.jpg	80
Bearbeiten Kopieren Löschen	100021	Datenbanken: Implementierungstechniken	9.99	82661438.jpg	70
Bearbeiten Kopieren Löschen	100023	MySQL 5 - kurz & gut	9.99	89721525A.jpg	100
Bearbeiten Kopieren Löschen	100020	MySQL - Das große Buch	9.99	23679192BT.jpg	70
Bearbeiten Kopieren Löschen	100001	Softwareentwicklung - Einstieg für Anspruchsvolle	14.99	82732851.jpg	1
Bearbeiten Kopieren Löschen	100024	PHP 5.3 & MySQL 5.1	19.99	21293244T.jpg	34
Bearbeiten Kopieren Löschen	100022	Datenbanken - Grundlagen und Design	19.99	82665529.jpg	60

Mit Gruppierung

```
SELECT * FROM artikel GROUP BY APreis ORDER BY APreis ASC
```

	ANr	AName	APreis	ABild	ABestand
Bearbeiten Kopieren Löschen	100002	PHP Referenz - Befehlskarte	1.00	93970129A.jpg	0
Bearbeiten Kopieren Löschen	100012	VBA mit Excel - easy	2.99	00000233A.jpg	50
Bearbeiten Kopieren Löschen	100016	Programmieren lernen für Kinder	9.99	82724415.jpg	80
Bearbeiten Kopieren Löschen	100001	Softwareentwicklung - Einstieg für Anspruchsvolle	14.99	82732851.jpg	1
Bearbeiten Kopieren Löschen	100013	CSS - Basis-Know-how	19.99	23351670BT.jpg	40
Bearbeiten Kopieren Löschen	100000	Handbuch der Java-Programmierung	29.99	82732874.jpg	20
Bearbeiten Kopieren Löschen	100019	SOA - Entwurfsprinzipien für serviceorientierte Ar...	39.99	24918456BT.jpg	80

SELECT * ist hier unsinnig, es machen nur die Spalten Sinn, die man auch betrachtet – hier `APreis` und `ABestand`

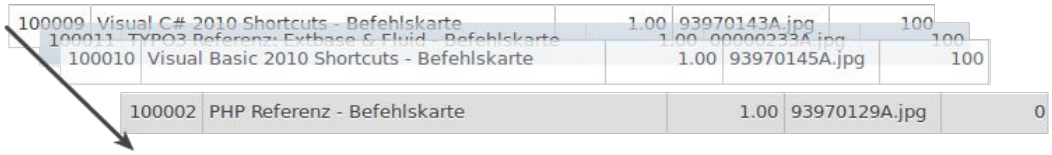
Warum macht die Wildcard Abfrage keinen Sinn?

```
SELECT * FROM artikel GROUP BY APreis ORDER BY APreis ASC
```

SELECT * ist hier unsinnig, es machen nur die Spalten Sinn, die man auch betrachtet – hier `APreis` und `Abestand`

Sonst erhält man unsinnige Ergebnisse:

100009	Visual C# 2010 Shortcuts - Befehlskarte	1.00	93970143A.jpg	100
100011	TYPO3 Referenz: Extbase & Fluid - Befehlskarte	1.00	00000233A.jpg	100
100010	Visual Basic 2010 Shortcuts - Befehlskarte	1.00	93970145A.jpg	100
100008	CSS Referenz - Befehlskarte	1.00	93970156A.jpg	100
100007	HTML Referenz - Befehlskarte	1.00	93970155A.jpg	100
100006	XML & DTD Referenz - Befehlskarte	1.00	93970135A.jpg	100
100005	C# .NET Referenz - Befehlskarte	1.00	93970130A.jpg	100
100004	Java Referenz - Befehlskarte	1.00	93970138A.jpg	100
100003	XML Schema Referenz - Befehlskarte	1.00	93970128A.jpg	70
100002	PHP Referenz - Befehlskarte	1.00	93970129A.jpg	0
100012	VBA mit Excel - easy	2.99	00000233A.jpg	50
100016	Programmieren lernen für Kinder	9.99	82724415.jpg	80
100021	Datenbanken: Implementierungstechniken	9.99	82661438.jpg	70
100023	MySQL 5 - kurz & gut	9.99	89721525A.jpg	100
100020	MySQL - Das große Buch	9.99	23679192BT.jpg	70
100001	Softwareentwicklung - Einstieg für	14.99	82732851.jpg	1



Alle Artikel mit dem Preis 1.00 werden "übereinander" gelegt. Sichtbar ist nur "die oberste Schicht", ein zufälliger Datensatz (in diesem Beispiel der letzte).

Während der Gruppierung kann kann aber z.B. mit SUM(ABestand) der Gesamtbestand einer Gruppe bestimmt werden. Oder mit AVG der Durchschnittspreis...

So klappts...

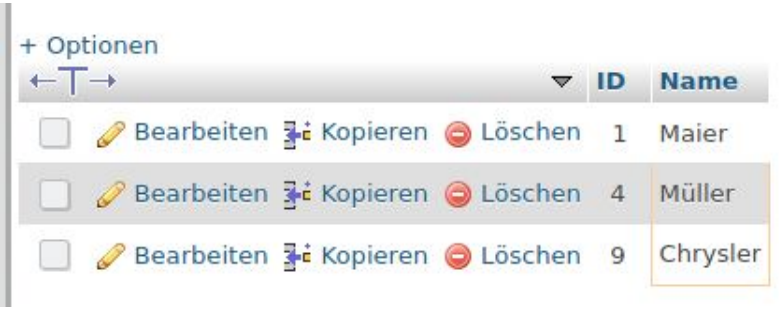
```
SELECT Apreis,SUM(ABestand)
FROM artikel
GROUP BY Apreis
ORDER BY APreis ASC ASC
```

APreis	SUM(ABestand)
1.00	870
2.99	50
9.99	320
14.99	1
19.99	686
29.99	197
39.99	80

Aliase

Mit dem Schlüsselwort **AS**. Kann man Aliase für Tabellen(spalten) erstellen:

```
SELECT KundenID AS ID, KundenName AS Name
FROM Kunden WHERE KundenName LIKE '%er';
```



- Kürzere SQL Statements
- Bessere Verständlichkeit

```
SELECT AngestelltenID AS ID, AVG(AngestelltenGehalt) AS Durchschnittsgehalt
FROM Mitarbeiter WHERE Angestelltengehalt > 2000 GROUP BY Abteilung;
```

Bedingungen für Gruppen

Liegt die Filterbedingung als Tabellenattribut vor, filtert man **vor der Gruppierung mit WHERE**

```
SELECT COUNT(AngestelltenID) AS Anzahl,
AngestelltenID AS ID,
AVG(AngestelltenGehalt) AS Durchschnittsgehalt
FROM Mitarbeiter WHERE AngestelltenGehalt > 1500 GROUP BY Abteilung
```

Anzahl	ID	Durchschnittsgehalt
1	6	1976.2299046875
2	9	2994.9400634765625
2	11	9498.6845703125
2	1	2040.2999877929688

Eine Filterbedingung für gruppiert Werte bringt man **nach der Gruppierung mit HAVING** an:

```
SELECT COUNT(AngestelltenID) AS Anzahl,
AngestelltenID AS ID,
AVG(AngestelltenGehalt) AS Durchschnittsgehalt
FROM Mitarbeiter
GROUP BY Abteilung HAVING Durchschnittsgehalt > 2000
```

Anzahl	ID	Durchschnittsgehalt
2	9	2994.9400634765625
2	11	9498.6845703125

Oder beides:

```
SELECT COUNT(AngestelltenID) AS Anzahl,
AngestelltenID AS ID,
AVG(AngestelltenGehalt) AS Durchschnittsgehalt
FROM Mitarbeiter
WHERE AngestelltenGehalt > 1500
GROUP BY Abteilung HAVING Durchschnittsgehalt > 2000
```

Anzahl	ID	Durchschnittsgehalt
2	9	2994.9400634765625
2	11	9498.6845703125
2	1	2040.2999877929688

Zusammenfassung Gruppierungen

Allgemeiner Aufbau

```
SELECT [DISTINCT] { spalten | * }
FROM              tabelle [alias] [,tabelle [alias]] …
[WHERE            {bedingung}]
[GROUP BY         spalten [HAVING {bedingung}]]
[ORDER BY         spalten [ASC | DESC]];
```

Auswertungsreihenfolge

SELECT	(Spaltenauswahl)
FROM	(Tabellenauswahl)
WHERE	(Zeilenauswahl bzw. Selektion)
GROUP BY	(Gruppierung)
HAVING	(Gruppenauswahl)
ORDER BY	(Sortierung)

Änderung (Manipulation) von Daten mit SQL

INSERT: Einfügen neuer Datensätze

```
INSERT INTO tabelle(spalte1,spalte2,...)
VALUES ('string1',integer2,...)
```

Optional, wenn man alle Felder der Tabelle im der Reihenfolge der Definition mit Werten befüllt.

UPDATE: Ändern bestehender Datensätze

```
UPDATE tabelle
SET    spalte1='wert1', spalte2='wert2',...
[WHERE bedingung]
```

DELETE: Löschen bestehender Datensätze

```
DELETE FROM tabelle
[WHERE      bedingung]
```

Vorsicht: **`DELETE FROM tabelle`** leert eine Tabelle.