Prof. DI Dr. Erich Gams

Datenmanipulation in SQL

Erweiterte Abfragen mit Gruppierungen

To start….

Please, close your laptops

and just

Übersicht ☛ Was lernen wir?

- Verschiedene Aggregatfunktionen
- Group by
- Having

Aggregatfunktion COUNT

- Wie viele Kunden gibt es? Wir zählen die Zeilen der Kundentabelle.

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

- Die Funktion COUNT(*), die hier angewandt wird, ist eine *Aggregatfunktion*.
- Sie zählt in diesem Fall die Anzahl der gefundenen Zeilen.

Aggregatfunktion COUNT

- Die COUNT-Funktion, auf eine Spalte angewandt, zählt die tatsächlich vorhandenen Werte (ohne NULL).

```
SELECT COUNT(bestelldatum), COUNT(lieferdatum)
FROM Bestellung;
```

COUNT(bestelldatum)	COUNT(lieferdatum)
4	2

- Ergebnistabelle = eine einzige Zeile.
- Die Funktion COUNT berechnet also aus allen Zeilen einer Tabelle genau einen Wert.

Aggregatfunktion COUNT

- Schließlich kann mit der COUNT-Funktion auch die Anzahl verschiedener Werte in einer Spalte ermittelt werden.
- Beispielsweise lassen wir mit folgender Anweisung die Anzahl verschiedener Postleitzahlen bei unseren Kunden anzeigen:

```
SELECT COUNT(DISTINCT plz)
FROM kunde;
```

Weitere Aggregatfunktionen

- In SQL sind ursprünglich fünf Aggregatfunktionen definiert, die dort als *set functions* bezeichnet werden:

COUNT	Anzahl von Tupeln, Werten oder verschiedenen Werten
SUM	Summe der Werte
AVG	Durchschnitt der Werte
MAX	Größter Wert
MIN	Kleinster Wert

Weitere Aggregatfunktionen

- Alle Funktionen lassen sich in einer einzigen SELECT-Anweisung unterbringen, wie das folgende Beispiel zeigt.
- Wir fragen nach der Anzahl aller Artikel, der Summe und dem Durchschnitt ihrer Einzelpreise sowie nach dem größten und kleinsten Einzelpreis.

```
SELECT
    COUNT(artikel_nr)AS ANZAHL,
    SUM(listenpreis) AS PREISSUMME,
    AVG(listenpreis) AS DURCHSCHN,
    MIN(listenpreis) AS KLEINSTER,
    MAX(listenpreis) AS GROESSTER
    FROM artikel;
```

ANZAHL	PREISSUMME	DURCHSCHN	KLEINSTER	GROESSTER
12	428,33	35,6941667	0,98	112,8

Nullmarken (=NULL) bei Aggregatfunktionen

- Nullmarken werden bei Anwendung der Aggregatfunktionen nicht berücksichtigt.
- Auswirkungen:
  - Summenbildung: Führt zu demselben Resultat, als wenn NULL die Zahl 0 repräsentierte.)
  - Minimum- Maximumbildung und Durchschnittsbildung: Nullmarken fallen ganz heraus.
- Bei leeren Ergebnismengen sind MIN, AVG, MAX nicht definiert.
- Die COUNT-Funktion, auf eine Spalte angewandt, zählt die tatsächlich vorhandenen Werte.
- Achtung: Bei Spalten, für die Nullmarken zulässig sind, kann deshalb das Ergebnis COUNT(spalte) kleiner sein als COUNT(*).

SELECT mit GROUP BY

- Die Aggregatfunktionen erlauben weitergehende Auswertungen, indem bestimmte Teilmengen der Tupel einer Relation zu Gruppen zusammengefasst werden.

- Das Zählen, die Durchschnitts- und Summenbildung, die Ermittlung von Minimal- und Maximalwerten kann beispielsweise so bezogen werden:
  - jeweils auf alle Bestellungen eines Kunden,
  - jeweils auf alle Positionen einer Bestellung,
  - auf alle Positionen, in denen ein Artikel bestellt oder berechnet wird,
  - auf alle Kunden einer Stadt.

SELECT mit GROUP BY

- Entscheidend für die Gruppenbildung sind gleiche Werte in einer bestimmten Spalte.
  - Für Bestellungen desselben Kunden ist der Wert der Spalte *kunden_nr* in *bestellung* immer gleich.
  - Bei Positionen, die denselben Artikel betreffen, ist der Wert der Spalte *artikel_nr* in der Tabelle *bestellposition* immer derselbe.

Beispiele

› Wie oft hat jeder Kunde bestellt?

Kunden_nr	Bestellnr
101	1234
103	3456
103	9876
105	4567

Beispiele

› Wie oft hat jeder Kunde bestellt?

```
SELECT kunden_nr, COUNT(*)
FROM Bestellung
GROUP BY kunden_nr;
```

Kunden nr	count(*)
101	1
103	2
105	1

Beispiele

- Für jeden Artikel ist die Anzahl aller Bestellungen zu ermitteln.

```
SELECT  artikel_nr, SUM (bestellmenge)
  FROM  bestellposition
  GROUP BY artikel_nr;
```

artikel_nr	SUM(bestellmenge)
K003	3
L002	16
K002	3
G001	6
L004	5
K004	12
G003	4
G002	16
K001	10
L003	25

Beispiele

- Wann war die jeweils letzte Bestellung der Kunden?

```
SELECT  kunden_nr, MAX (bestelldatum)
  FROM  bestellung
  GROUP BY kunden_nr;
```

kunden_nr	MAX(bestelldatum)
103	2008-05-15
101	2008-04-28
105	2008-05-12

SELECT mit GROUP BY und HAVING

- Die Gruppierung erzeugt eine Relation, in der Attribute auftreten, die für jede Gruppe einen Wert haben.
- Bis auf die Gruppierungsspalten selbst, deren Werte sich in der Originaltabelle befinden, werden die Werte erst zur Laufzeit der Abfrage berechnet.
- Mit der zusätzlichen HAVING-Klausel kann nun eine Selektion *nach* der Gruppenbildung vorgenommen werden.
- Das bedeutet, es erfolgt eine Selektion unter den Gruppen.

- Für jeden Artikel hatten wir die Anzahl aller Bestellungen so ermittelt:

```
SELECT artikel_nr, SUM (bestellmenge)
  FROM  bestellposition
  GROUP BY artikel_nr;
```

artikel_nr	SUM(bestellmenge)
K003	3
L002	16
K002	3
G001	6
L004	5
K004	12
G003	4
G002	16
K001	10
L003	25

HAVING

- Die Auswertung zeigt, dass es Artikel gibt, die mehr als zehnmal bestellt wurden, und solche, die weniger oft geordert wurden.
- Wollen wir nur Artikel mit Bestellmengen von über zehn berücksichtigen, so muss die obige SQL-Anweisung um eine HAVING-Klausel ergänzt werden.

HAVING

```
... HAVING SUM(bestellmenge) > 10
```

Das Ergebnis:

```
SELECT  artikel_nr, SUM (bestellmenge)
    FROM      bestellposition
    GROUP BY artikel_nr
    HAVING SUM(bestellmenge) > 10;
```

artikel_nr	SUM(bestellmenge)
G002	16
K004	12
L002	16
L003	25

HAVING

- Die Grundform der HAVING-Klausel in Verbindung mit einer GROUP BY-Klausel lautet:

```
GROUP BY spaltenliste
HAVING bedingung
```

- Da die HAVING-Klausel die Ergebnisse der Gruppierung selektiert, enthält sie meist selbst eine Aggregatfunktion.
- Beispiele:

```
... HAVING COUNT(*) > 1
... HAVING SUM(LIEFERMENGE) < SUM(BESTELLMENGE)
```

Reihenfolge

1. Zunächst wird mit der WHERE-Klausel eine Selektion auf der Basistabelle ausgeführt.
2. Die verbleibenden Tupel werden gruppiert.
3. Zum Schluss werden die Gruppen selektiert.

WHERE

GROUP BY

HAVING

Beispiel

- Die Kombination von WHERE und HAVING zeigt das folgende Beispiel, in dem die Anzahl der Bestellpositionen ermittelt wird, in denen ein Artikel mehr als einmal auftritt.
- Vor der Gruppierung werden aber die Tupel selektiert, bei denen die Artikelnummer mit 'K' anfängt.

```
SELECT artikel_nr, SUM(bestellmenge)
FROM bestellposition
WHERE artikel_nr LIKE ‘K%‘
GROUP BY artikel_nr
HAVING SUM(bestellmenge)>1;
```

Beispiel

- <u>Frage:</u> Könnte ich die `sum(bestellmenge)>1` Einschränkung auch in einer `where` Klausel schreiben?
- <u>Antwort:</u> Nein! Gibt einen Fehler!
- Die Summe bezieht sich immer auf eine Gruppierung, da die Artikelnummer mit angegeben ist!