GROUP BY, HAVING und Sichten

Tutorübungen 09/33 zu Grundlagen: Datenbanken (WS 14/15)

Michael Schwarz

Technische Universität München

11.11 / 12.11.2014

Eine Tabelle

studenten			
name	**fach**	**uni**	**sem**
Max	Informatik	TUM	3
Gabi	Informatik	LMU	4
Flo	Geographie	LMU	1
Natalie	Geographie	TUM	3
Lena	Germanistik	LMU	3
Bastian	Mathematik	HM	5
John	Informatik	LMU	10

Eine Abfrage

Wie viele Studenten pro Uni ?
⇒ Aggregation nur auf einzelne Gruppe und nicht auf ganze Tabelle

Lösung

Hier für bietet SQL **GROUP BY** an.

Eine Tabelle

studenten			
name	**fach**	**uni**	**sem**
Max	Informatik	TUM	3
Gabi	Informatik	LMU	4
Flo	Geographie	LMU	1
Natalie	Geographie	TUM	3
Lena	Germanistik	LMU	3
Bastian	Mathematik	HM	5
John	Informatik	LMU	10

Eine Abfrage

Wie viele Studenten pro Uni ?
⇒ Aggregation nur auf einzelne Gruppe und nicht auf ganze Tabelle

Lösung

Hier für bietet SQL **GROUP BY** an.

Eine Tabelle

studenten			
name	**fach**	**uni**	**sem**
Max	Informatik	TUM	3
Gabi	Informatik	LMU	4
Flo	Geographie	LMU	1
Natalie	Geographie	TUM	3
Lena	Germanistik	LMU	3
Bastian	Mathematik	HM	5
John	Informatik	LMU	10

Eine Abfrage

Wie viele Studenten pro Uni ?
⇒ Aggregation nur auf einzelne Gruppe und nicht auf ganze Tabelle

Lösung

Hier für bietet SQL **GROUP BY** an.

GROUP BY

studenten			
name	**fach**	**uni**	**sem**
Max	Informatik	TUM	3
Gabi	Informatik	LMU	4
Flo	Geographie	LMU	1
Natalie	Geographie	TUM	3
Lena	Germanistik	LMU	3
Bastian	Mathematik	HM	5
John	Informatik	LMU	10

Abfrage

```
SELECT uni, COUNT(*) FROM studenten GROUP BY uni;
```

GROUP BY

studenten			
name	**fach**	**uni**	**sem**
Max	Informatik	TUM	3
Gabi	Informatik	LMU	4
Flo	Geographie	LMU	1
Natalie	Geographie	TUM	3
Lena	Germanistik	LMU	3
Bastian	Mathematik	HM	5
John	Informatik	LMU	10

Abfrage

```
SELECT uni, COUNT(*) FROM studenten GROUP BY uni;
```

GROUP BY

name	fach	uni	sem
Gabi	Informatik	LMU	4
Flo	Geographie	LMU	1
Lena	Germanistik	LMU	3
John	Informatik	LMU	10

name	fach	uni	sem
Max	Informatik	TUM	3
Natalie	Geographie	TUM	3

name	fach	uni	sem
Bastian	Mathematik	HM	5

Abfrage

```
SELECT uni, COUNT(*) FROM studenten GROUP BY uni;
```

GROUP BY

uni	count
TUM	2
LMU	4
HM	1

Abfrage

```
SELECT uni, COUNT(*) FROM studenten GROUP BY uni;
```

Aber Vorsicht...

... im SELECT nur Aggregation von Attributen und Attribute nach denen gruppiert wurde.

GROUP BY

uni	count
TUM	2
LMU	4
HM	1

Abfrage

```
SELECT uni, COUNT(*) FROM studenten GROUP BY uni;
```

Aber Vorsicht...

... im SELECT nur Aggregation von Attributen und Attribute nach denen gruppiert wurde.

Was nicht funktioniert

studenten			
name	**fach**	**uni**	**sem**
Max	Informatik	TUM	3
Gabi	Informatik	LMU	4
Flo	Geographie	LMU	1
Natalie	Geographie	TUM	3
Lena	Germanistik	LMU	3
Bastian	Mathematik	HM	5
John	Informatik	LMU	10

Abfrage

```
SELECT uni, fach, COUNT(*) FROM studenten
GROUP BY uni;
```

Was nicht funktioniert:

name	fach	uni	sem
Gabi	Informatik	LMU	4
Flo	Geographie	LMU	1
Lena	Germanistik	LMU	3
John	Informatik	LMU	10

name	fach	uni	sem
Max	Informatik	TUM	3
Natalie	Geographie	TUM	3

name	fach	uni	sem
Bastian	Mathematik	HM	5

Abfrage

```
SELECT uni, fach, COUNT(*) FROM studenten
GROUP BY uni;
```

KEINE gültige Anfrage!

Wie viele Stundenten studieren ein Fach an einer Uni ?

```
SELECT uni, fach, COUNT(*) FROM studenten
GROUP BY uni, fach;
```

Was nicht funktioniert:

name	fach	uni	sem
Gabi	Informatik	LMU	4
Flo	Geographie	LMU	1
Lena	Germanistik	LMU	3
John	Informatik	LMU	10

name	fach	uni	sem
Max	Informatik	TUM	3
Natalie	Geographie	TUM	3

name	fach	uni	sem
Bastian	Mathematik	HM	5

Abfrage

```
SELECT uni, fach, COUNT(*) FROM studenten
GROUP BY uni;
```

KEINE gültige Anfrage!

Wie viele Stundenten studieren ein Fach an einer Uni ?

```
SELECT uni, fach, COUNT(*) FROM studenten
GROUP BY uni, fach;
```

Was nicht funktioniert:

name	fach	uni	sem
Gabi	Informatik	LMU	4
Flo	Geographie	LMU	1
Lena	Germanistik	LMU	3
John	Informatik	LMU	10

name	fach	uni	sem
Max	Informatik	TUM	3
Natalie	Geographie	TUM	3

name	fach	uni	sem
Bastian	Mathematik	HM	5

Abfrage

```
SELECT uni, fach, COUNT(*) FROM studenten
GROUP BY uni;
```

KEINE gültige Anfrage!

Wie viele Stundenten studieren ein Fach an einer Uni ?

```
SELECT uni, fach, COUNT(*) FROM studenten
GROUP BY uni, fach;
```

Allgemeines

WHERE wird vor der Gruppierung ausgeführt, HAVING dannach.

Beispiel 1

Wie viele Stundenten sind je Uni mindestens im 3. Semester ?

```
SELECT uni, COUNT(*) FROM studenten
WHERE sem >= 3
GROUP BY uni;
```

Beispiel 2

Welche Unis haben mehr als 2 Studenten ?

```
SELECT uni FROM studenten
GROUP BY uni
HAVING COUNT(*) > 2;
```

Allgemeines

WHERE wird vor der Gruppierung ausgeführt, HAVING dannach.

Beispiel 1

Wie viele Stundenten sind je Uni mindestens im 3. Semester ?

```
SELECT uni, COUNT(*) FROM studenten
WHERE sem >= 3
GROUP BY uni;
```

Beispiel 2

Welche Unis haben mehr als 2 Studenten ?

```
SELECT uni FROM studenten
GROUP BY uni
HAVING COUNT(*) > 2;
```

Allgemeines

WHERE wird vor der Gruppierung ausgeführt, HAVING dannach.

Beispiel 1

Wie viele Stundenten sind je Uni mindestens im 3. Semester ?

```
SELECT uni, COUNT(*) FROM studenten
WHERE sem >= 3
GROUP BY uni;
```

Beispiel 2

Welche Unis haben mehr als 2 Studenten ?

```
SELECT uni FROM studenten
GROUP BY uni
HAVING COUNT(*) > 2;
```

Beispiel 3

Welche Unis haben mehr als 2 Studenten, die im 3. oder einem höheren Semester sind ?

```
SELECT uni FROM studenten
WHERE sem >= 3
GROUP BY uni
HAVING COUNT(*) > 2;
```

Sichten

Allgemein

- “Virtuelle Relation”
- Anfrage wird zwischengespeichert, Ergebnis aber jedes mal aktuell berechnet.
- Zugriff wie auf jede andere Relation

SQL

```
CREATE VIEW meinview AS
  SELECT ... ;

SELECT * FROM meinview;
```

WITH

Vllt. möchte man eine Sicht nicht speichern (oder darf es nicht). Alternativ kann man hier WITH verwenden

```
WITH meinedaten AS (SELECT ...)
SELECT ... FROM meinedaten,...;
```

GROUP BY, HAVING und Sichten

Tutorübungen 09/33 zu Grundlagen: Datenbanken (WS 14/15)

Michael Schwarz

Technische Universität München

11.11 / 12.11.2014