Data Aggregation

How to get data insights?

SoftUni Team

Technical Trainers

Software University

http://softuni.bg

Table of Content

Grouping

Consolidating data based on criteria

Aggregate Functions

COUNT, SUM, MAX, MIN, AVG…

Having

Using predicates while grouping

Questions

sli.do

#JavaDB

Grouping

Consolidating data based on criteria

Grouping

- Grouping allows taking data into separate groups based on a common property

Grouping column

employee	department_name	salary
Adam	Database Support	5,000
John	Database Support	15,000
Jane	Application Support	10,000
George	Application Support	15,000
Lila	Application Support	5,000
Fred	Software Support	15,000

Can be aggregated

GROUP BY

- With **GROUP BY** you can get each separate group and use an "aggregate" function over it (like Average, Min or Max):

```
SELECT e.`job_title`, count(employee_id)
  FROM `employees` AS e
GROUP BY e.`job_title`;
```

Grouping Columns

DISTINCT

- With **DISTINCT** you will get all unique values:

```
SELECT DISTINCT e.`job_title`
  FROM `employees` AS e;
```

Unique Values

Problem: Departments Total Salaries

- Write a query which prints the total sum of salaries for each department in the soft_uni database
  - Order them by DepartmentID (ascending)

employee	department_name	salary
Adam	Database Support	5,000
John	Database Support	15,000
Jane	Application Support	10,000
George	Application Support	15,000
Lila	Application Support	5,000
Fred	Software Support	15,000

department_id	total_salary
1	20,000
2	30,000
3	15,000

Check your solution here: https://judge.softuni.bg/Contests/Practice/Index/296#18

Solution: Departments Total Salaries

Grouping Column

```
SELECT e.`department_id`,
  SUM(e.`salary`) AS 'Total Salary'
FROM `employees` AS e
GROUP BY e.`department_id`
ORDER BY e.`department_id`;
```

New Column Alias

Table Alias

Grouping Columns

Check your solution here: https://judge.softuni.bg/Contests/Practice/Index/296#18

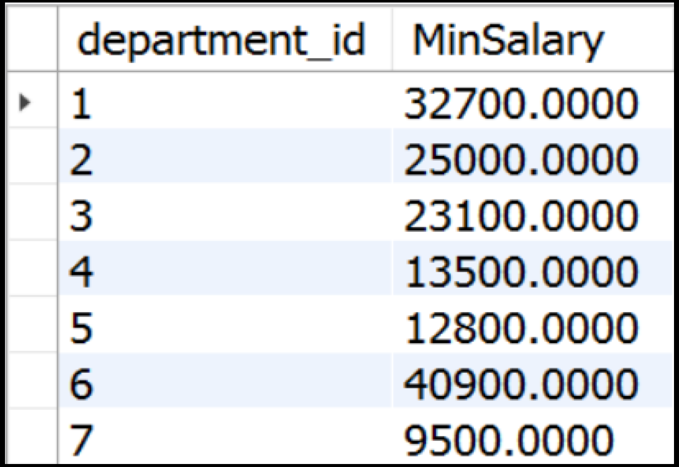
Aggregate Functions

COUNT, SUM, MAX, MIN, AVG…

Aggregate Functions

- Used to operate over one or more groups performing data analysis on every one
  - MIN, MAX, AVG, COUNT etc.
- They usually ignore **NULL** values

```sql
SELECT e.`department_id`,
 MIN(e.`salary`) AS 'MinSalary'
FROM `employees` AS e
GROUP BY e.`department_id`;
```



department_id	MinSalary
1	32700.0000
2	25000.0000
3	23100.0000
4	13500.0000
5	12800.0000
6	40900.0000
7	9500.0000

COUNT

- **COUNT** - counts the values (not nulls) in one or more columns based on grouping criteria

employee	department_name	salary
Adam	Database Support	5,000
John	Database Support	15,000
Jane	Application Support	10,000
George	Application Support	15,000
Lila	Application Support	5,000
Fred	Software Support	15,000

department_name	SalaryCount
Database Support	2
Application Support	3
Software Support	1

COUNT Syntax

- Note that we when we use **COUNT** we will ignore any employee with **NULL** salary.

Grouping Column

New Column Alias

```
SELECT e.`department_id`,
  COUNT(e.`salary`) AS 'Salary Count'
FROM `employees` AS e
GROUP BY e.`department_id`;
```

Grouping Columns

SUM

- **SUM** - sums the values in a column

employee	department_name	salary
Adam	Database Support	5,000
John	Database Support	15,000
Jane	Application Support	10,000
George	Application Support	15,000
Lila	Application Support	5,000
Fred	Software Support	15,000

department_name	total_salary
Database Support	20,000
Application Support	30,000
Software Support	15,000

SUM Syntax

- If any department has no salaries **NULL** will be displayed.

Grouping Column

New Column Alias

```
SELECT e.`department_id`,
  SUM(e.`salary`) AS 'TotalSalary'
FROM `employees` AS e
GROUP BY e.`department_id`;
```

Table Alias

Grouping Columns

MAX

- **MAX -** takes the maximum value in a column.

employee	department_name	salary
Adam	Database Support	5,000
John	Database Support	15,000
Jane	Application Support	10,000
George	Application Support	15,000
Lila	Application Support	5,000
Fred	Software Support	15,000

department_name	max_salary
Database Support	20,000
Application Support	30,000
Software Support	15,000

MAX Syntax

Grouping Column

```
SELECT e.`department_id`,
  MAX(e.`salary`) AS 'MaxSalary'
FROM `employees` AS e
 GROUP BY e.`department_id`;
```

New Column Alias

Table Alias

Grouping Columns

MIN

- **MIN** takes the minimum value in a column.

employee	department_name	salary
Adam	Database Support	5,000
John	Database Support	15,000
Jane	Application Support	10,000
George	Application Support	15,000
Lila	Application Support	5,000
Fred	Software Support	15,000

department_name	min_salary
Database Support	5,000
Application Support	5,000
Software Support	15,000

MIN Syntax

Grouping Column

```
SELECT e.`department_id`,
  MIN(e.`salary`) AS 'MinSalary'
FROM `employees` AS e
GROUP BY e.`department_id`;
```

New Column Alias

Table Alias

Grouping Columns

AVG

- **AVG** calculates the average value in a column.

employee	department_name	salary
Adam	Database Support	5,000
John	Database Support	15,000
Jane	Application Support	10,000
George	Application Support	15,000
Lila	Application Support	5,000
Fred	Software Support	15,000

department_name	average_salary
Database Support	10,000
Application Support	10,000
Software Support	15,000

Demo: AVG Syntax

Grouping Column

New Column Alias

Table Alias

Grouping Columns

```
SELECT e.`department_id`,
  AVG(e.`salary`) AS 'AvgSalary'
FROM `employees` AS e
GROUP BY e.`department_id`;
```

Having

Using predicates while grouping

Having Clause

- The **HAVING** clause is used to filter data based on aggregate values.
  - We cannot use it without grouping before that
- Any Aggregate functions in the "**HAVING**" clause and in the "**SELECT**" statement are executed one time only
- Unlike **HAVING**, the **WHERE** clause filters rows before the aggregation

Having Clause: Example

- Filter departments which have total salary more or equal 15,000.

Aggregated value

employee	department_name	salary	Total Salary
Adam	Database Support	5,000	20,000
John	Database Support	15,000	
Jane	Application Support	10,000	10,000
George	Application Support	15,000	
Lila	Application Support	5,000	
Fred	Software Support	15,000	15,000

department_name	average_salary
Database Support	10,000
Software Support	15,000

HAVING Syntax

Aggregate Function

Grouping Column

```
SELECT e.`department_id`,
  SUM(e.salary) AS 'TotalSalary'
FROM `employees` AS e
GROUP BY e.`department_id`
HAVING `TotalSalary`< 250000;
```

New Column Alias

Grouping Columns

Having Predicate

Summary

- Grouping
- Aggregate Functions
- Having

```
SELECT
   SUM(e.`salary) AS 'TotalSalary'
FROM `employees` AS e
GROUP BY e.`department_id`
HAVING SUM(e.`salary`) < 250000;
```

Data Aggregation

Questions?

https://softuni.bg/courses/databases-basics-mysql

License

- This course (slides, examples, demos, videos, homework, etc.) is licensed under the "Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International" license

- Attribution: this work may contain portions from
  - "Databases" course by Telerik Academy under CC-BY-NC-SA license

Free Trainings @ Software University

- Software University Foundation – softuni.org
- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg