

Week-8



Introduction to Database Systems

Shah Nawaz

The header features a white background on the left with various school supplies: a pair of blue-handled scissors, a white eraser, a red pencil, a small blue sharpener, and a pair of compasses. To the right, a blue background contains a faint, semi-transparent image of a protractor.

Introduction

- An aggregate function performs a calculation on a set of values of a column, and returns a single value
- Built-in aggregate functions
COUNT, SUM, MAX, MIN, and AVG
- Functions can be used in the SELECT clause or in a HAVING clause



SUM, MAX, MIN, AVG

- Find the sum of the salaries of all employees, the maximum salary, the minimum salary, and the average salary.

```
SELECT SUM (Salary), MAX (Salary), MIN (Salary), AVG(Salary)  
FROM EMPLOYEE;
```

Sum (Salary)	Max(Salary)	Min (Salary)	Avg (Salary)
986550	55000	25000	55600



COUNT

Find the total number of employees.

```
SELECT count(ssn)  
FROM EMPLOYEE;
```

The above query will ignore nulls but not duplicates.

```
SELECT count(*)  
FROM EMPLOYEE;
```

The above query will not ignore nulls & duplicates. count(*) counts the rows.

Count(ssn)
8

Count(*)
10



How Count() handles null values?

Field1	Field2	Field3
1	1	1
NULL	NULL	NULL
2	2	NULL
1	3	1

Then

```
SELECT COUNT(*), COUNT(Field1), COUNT(Field2), COUNT(DISTINCT Field3)
FROM Table1
```

Output Is:

```
COUNT(*) = 4; -- count all rows, even null/duplicates

-- count only rows without null values on that field
COUNT(Field1) = COUNT(Field2) = 3

COUNT(Field3) = 2
COUNT(DISTINCT Field3) = 1 -- Ignore duplicates
```

DISTINCT

Count the number of distinct salary values in the database

```
SELECT COUNT (DISTINCT CUST_CODE) AS  
"Number of Employees"  
FROM ORDERS  
Where Agent_code='A010' OR  
Agent_code='A009';3
```

ORD_NUM	AGENT_CODE	CUST_CODE
200101	A008	C00001
200114	A008	C00002
200122	A004	C00003
200134	A005	C00004
200106	A002	C00005
200104	A004	C00006
200135	A010	C00007
200119	A010	C00007
200107	A010	C00007
200108	A004	C00008
200121	A004	C00008
200120	A002	C00009
200133	A002	C00009
200128	A002	C00009
200116	A009	C00010
200109	A010	C00011
200102	A012	C00012
200131	A012	C00012

** for better understand
the table have been arranged on
cust_code

Number of employees
12

result

count once

Group By

- Aggregate functions are often used with the GROUP BY clause
- GROUP BY Clause is used to collect data from multiple records and group the result by one or more column
- **Example:** "find the number of authors in each country".

SELECT column_name(s)
FROM table_name
WHERE condition
GROUP BY column_name(s)
ORDER BY column_name(s);

Select Country, count(*)
From Author
Group by country;

Here is the Output

aut_id	country
AUT011	Australia
AUT013	Australia
AUT009	Brazil
AUT002	Canada
AUT012	Canada
AUT005	Germany
AUT004	India
AUT007	UK
AUT014	UK
AUT003	UK
AUT001	UK
AUT010	USA
AUT008	USA
AUT006	USA
AUT015	USA

country	COUNT(*)
Australia	2
Brazil	1
Canada	2
Germany	1
India	1
UK	4
USA	4

group of country



Example:

For each department, retrieve the department number, the number of employees in the department, and their average salary.

```
SELECT    Dno, COUNT (*), AVG (Salary)
FROM      EMPLOYEE
GROUP BY Dno;
```

Dno	Count(*)	Avg(Salary)
4	3	30000
5	4	31000
1	1	55000



Order of execution

ORDER	CLAUSE	FUNCTION
1	from	Choose and join tables to get base data.
2	where	Filters the base data.
3	group by	Aggregates the base data.
4	having	Filters the aggregated data.
5	select	Returns the final data.
6	order by	Sorts the final data.
7	limit	Limits the returned data to a row count.



Relational Algebra



γ (Aggregation)

- γ (**gamma**) is used for operations like SUM, AVG, COUNT, etc., often with grouping.
- It produces a relation where the result of the aggregation is stored as a new attribute.

Example:

```
SELECT SUM (Salary) As TotalSalary FROM EMPLOYEE;
```

In relational algebra, the expression for the above SQL query can be represented using the γ (**gamma**) operator to perform the summation:

```
 $\gamma$ SUM(Salary) AS TotalSalary (EMPLOYEE)
```

Pronunciation: "Gamma: Sum of Salary as TotalSalary, applied to the EMPLOYEE relation."

Explanation:

- γ represents the Group By operation.
- "SUM(Salary) AS TotalSalary" inside γ indicates that we are performing the sum aggregation on the "Salary" attribute and renaming the result as "TotalSalary".
- The relation involved is "EMPLOYEE".

Example:

SELECT MAX (Salary) As MaximumSalary FROM EMPLOYEE;

The relational algebra expression for the above SQL query can be represented using the γ (gamma) operator to perform the maximum aggregation:

γ MAX(Salary) AS MaximumSalary(EMPLOYEE)

Pronounce: "Gamma: Maximum of Salary as MaximumSalary, applied to the EMPLOYEE relation."

Explanation:

- γ represents the Group By operation.
- "MAX(Salary) AS MaximumSalary" inside γ indicates that we are performing the maximum aggregation on the "Salary" attribute and renaming the result as "MaximumSalary".
- The relation involved is "EMPLOYEE".

Example:

```
SELECT MIN (Salary) AS MinimumSalary FROM EMPLOYEE;
```

The relational algebra expression for the above SQL query can be represented using the γ (gamma) operator to perform the maximum aggregation:

```
 $\gamma$ MIN(Salary) AS MinimumSalary(EMPLOYEE)
```

Pronounce: "Gamma: Minimum of Salary as MinimumSalary, applied to the EMPLOYEE relation."

Explanation:

- γ represents the Group By operation.
- "MIN(Salary) AS MinimumSalary" inside γ indicates that we are performing the minimum aggregation on the "Salary" attribute and renaming the result as "MinimumSalary".
- The relation involved is "EMPLOYEE".

Example:

SELECT AVG (Salary) As AverageSalary FROM EMPLOYEE;

The relational algebra expression for the above SQL query can be represented using the γ (gamma) operator to perform the maximum aggregation:

γ AVG(Salary) AS AverageSalary(EMPLOYEE)

Pronounce: "Gamma: Average of Salary as AverageSalary, applied to the EMPLOYEE relation."

Explanation:

- γ represents the Group By operation.
- "MIN(Salary) AS AverageSalary" inside γ indicates that we are performing the Average aggregation on the "Salary" attribute and renaming the result as "AverageSalary".
- The relation involved is "EMPLOYEE".

```
SELECT COUNT (CUST_CODE) AS “Total_Customers”  
FROM ORDERS;
```

```
γ COUNT (CUST_CODE) AS Total_Customers(ORDERS)
```

Pronunciation: Gamma count customer code as total customers from orders table.

Explanation:

- γ:** Pronounced as "Gamma."
- COUNT(CUST_CODE):** Say "Count customer code."
- AS Total_Customers:** Say "as total customers."
- (ORDERS):** Say "of orders."

The header features a blue gradient background. On the left, there is a vertical strip showing school supplies: a pair of blue-handled scissors, a white eraser, a red pencil, a blue pen, and a pair of compasses. On the right, the text 'COUNT - DISTINCT' is written in white, bold, sans-serif font. A faint, semi-transparent image of a protractor is visible in the background.

COUNT - DISTINCT

```
SELECT COUNT (Distinct (CUST_CODE)) AS "Total_Customers"  
FROM ORDERS;
```

R.A:

```
γ COUNT (DISTINCT CUST_CODE) AS Total_Customers(ORDERS)
```

Pronunciation: " Gamma count distinct customer code as total customers from orders table"

Explanation:

- γ:** Pronounced as "Gamma."
- COUNT(DISTINCT CUST_CODE):** Say "Count distinct customer code."
- AS Total_Customers:** Say "as total customers."
- (ORDERS):** Say "of orders."



Group By

Example:

```
SELECT CustomerID, COUNT(*) AS TotalOrders  
FROM Orders GROUP BY CustomerID;
```

R.A:

```
γCustomerID, COUNT CustomerID AS TotalOrders (Orders)
```

Pronunciation: "Projection of 'CustomerID' and 'OrderCount' from the result of grouping Orders by 'CustomerID' and applying the COUNT aggregate function, with the result aliased as 'OrderCount'."

Explanation:

- **γ:** Pronounced as "Gamma." represents the grouping and aggregation operation.
- **CustomerID:** Specifies that the grouping is done based on the CustomerID attribute.
- **COUNT(*) → OrderCount:** Applies the COUNT(*) aggregate function to count the number of orders for each group and names the result as OrderCount.
- **Orders:** Refers to the input relation (table).

Practice

Group By Subject count ?
Group By Year Count?

SUBJECT	YEAR	NAME
English	1	Harsh
English	1	Pratik
English	1	Ramesh
English	2	Ashish
English	2	Suresh
Mathematics	1	Deepak
Mathematics	1	Sayan

The header features a blue gradient background. On the left, there is a collection of school supplies: a pair of blue-handled scissors, a white eraser, a red pencil, a blue sharpener, and a pair of compasses. On the right, the word "Practice:" is written in white. A faint, semi-transparent image of a protractor is visible in the background.

Practice:

- For each project, retrieve the project number, the project name, and the number of employees who work on that project.

```
SELECT
    p.project_number,
    p.project_name,
    COUNT(e.employee_id) AS num_employees
FROM
    projects p
JOIN
    employees e ON p.project_number = e.project_number
GROUP BY
    p.project_number, p.project_name;
```

The header features a white background on the left with various school supplies: a pair of blue-handled scissors, a white eraser, a red pencil, a small blue sharpener, and a pair of compasses. On the right, there is a blue gradient background with a faint, semi-transparent image of a protractor.

Practice:

- For each project, retrieve the project number, the project name, and the number of employees who work on that project.



- For each project, retrieve the project number, the project name, and the number of employees from department 5 who work on the project.



- For each project, retrieve the project number, the project name, and the number of employees from department 5 who work on the project.

Select pnumber, pname, count(essn)

From (project join workson on pno= pnumber) join employee on essn =ssn

Where dno = 5

group by pnumber;

The header features a blue gradient background. On the left, there is a collection of school supplies: a pair of blue-handled scissors, a white eraser, a red pencil, a blue sharpener, and a pair of compasses. On the right, the text 'Practice Queries' is displayed in a white, sans-serif font. A faint, semi-transparent image of a protractor is visible in the background behind the text.

Practice Queries

1. Show average salary of all employees.
2. Show total number of departments.
3. Show sum of hours spent on all projects.
4. Show average salary of each department if average salary is greater than 40,000.
5. Show minimum salary of each department if minimum salary is less than 20,000.
6. Show total number of employees of each department with department name.
7. Show number of dependent of each employee.
8. Show count of projects of all departments in ascending order by department name.
9. Show count of total locations of each department which are located in Lahore.
10. For each project show project name and total number of employees working on it.



Thank You all!