

# 4.3.3 Assignment 1:

## Aggregate Functions

**Scenario 1:** Using the customers table from mysqltutorial.org, write an SQL statement to find the total credit limit of all customers.

**Analysis:** This scenario involves calculating the total credit limit across all customers. The SUM aggregate function is used to add up the credit limits of all customers.


Table: customers

Column: creditLimit

Aggregate Function: SUM

Query:

```
SELECT SUM(creditLimit) AS totalCreditLimit FROM customers;
```



The screenshot shows a web-based SQL query editor. At the top, there is a tab labeled 'SQL QUERY'. Below it, the query is entered in a text area: `1 SELECT SUM(creditLimit) AS totalCreditLimit FROM customers;` and `2` on the next line. Below the text area are four buttons: 'Execute' (green), 'Clear', 'Beautify', and 'Minify'. Below the buttons is a tab labeled 'RESULT'. Under the 'RESULT' tab, the query result is displayed in a table with one column named 'totalCreditLimit' and one row with the value '8254400.00'.

totalCreditLimit
8254400.00

**Scenario 2:** Using the customers table from mysqltutorial.org, write an SQL statement to find the average credit limit of all customers.

**Analysis:** This scenario calculates the average credit limit of all customers. The AVG aggregate function is used to find the average credit limit.

Table: customers

Column: creditLimit

Aggregate Function: AVG

Query:

SELECT AVG(creditLimit) AS averageCreditLimit FROM customers;

The screenshot shows a web-based SQL editor interface. At the top, there is a tab labeled 'SQL QUERY'. Below it, a text area contains the following SQL query:

```
1 SELECT AVG(creditLimit) AS averageCreditLimit FROM customers;
2
```

Below the query editor, there is a row of buttons: 'Execute' (with a play icon), 'Clear' (with an eraser icon), 'Beautify' (with a magic wand icon), and 'Minify' (with a code icon). Below these buttons is a tab labeled 'RESULT'. Under the 'RESULT' tab, the query results are displayed in a table with two rows:

averageCreditLimit
67659.016393

**Scenario 3:** Using the products table from [mysqltutorial.org](https://www.mysqltutorial.org), write an SQL statement to find how many product lines are being offered by the company.

Analysis: This scenario counts the number of unique product lines offered by the company. The COUNT(DISTINCT ...) is used to count the distinct product lines.

Table: products

Column: productLine

Aggregate Function: COUNT(DISTINCT productLine)

Query:

SELECT COUNT(DISTINCT productLine) AS totalProductLines FROM products;

SQL QUERY

1

2

3

```
SELECT COUNT(DISTINCT productLine) AS totalProductLines FROM products;
```

▶ Execute

✍ Clear

✍ Beautify

✂ Minify

RESULT

totalProductLines
7

**Scenario 4:** Using the order\_details table from mysqltutorial.org, write an SQL statement to get the highest quantity of orders.

**Analysis** This scenario aims to find the highest quantity of orders.

The MAX aggregate function is used to determine the maximum quantity ordered.

Table: order\_details

Column: quantityOrdered

Aggregate Function: MAX

Query:

```
SELECT MAX(quantityOrdered) AS highestQuantity FROM order_details;
```

SQL QUERY

```
1 SELECT MAX(quantityOrdered) AS highestQuantity FROM orderdetails;
2
3
4
5
```

▶ Execute

✎ Clear

✎ Beautify

✎ Minify

RESULT

highestQuantity
97

**Scenario 5:** Using the payments table from mysqltutorial.org, write an SQL statement to get the minimum amount paid by the customer and the check number.

Analysis: This scenario retrieves the minimum amount paid by the customer along with the corresponding check number.

The MIN aggregate function is used to find the minimum payment amount.

Table: payments

Column: amount, checkNumber

Aggregate Function: MIN

Query:

```
SELECT MIN(amount) AS minimumAmountPaid, checkNumber FROM payments;
```

SQL QUERY

1

2

3

4

5

```
SELECT MIN(amount) AS minimumAmountPaid, checkNumber FROM payments;
```

▶ Execute

✕ Clear

✎ Beautify

✕ Minify

RESULT

minimumAmountPaid	checkNumber
615.45	HQ336336

**Scenario 6:** Using the orders table from [mysqldata.org](https://mysqldata.org/), write an SQL statement to get the number of orders shipped beginning 2003-01-10 and onwards.

Analysis: This scenario counts the number of orders shipped beginning from January 10, 2003. The COUNT aggregate function is used to calculate the total count.

Table: orders

Column: shippedDate

Condition: shippedDate >= '2003-01-10'

Aggregate Function: COUNT

Query:

```
SELECT COUNT(*) AS numberOfOrders FROM orders WHERE shippedDate >= '2003-01-10';
```

SQL QUERY

1

2

```
SELECT COUNT(*) AS numberOfOrders FROM orders WHERE shippedDate >= '2003-01-10';
```

▶ Execute

✎ Clear

✎ Beautify

✎ Minify

RESULT

numberOfOrders
312

**Scenario 7:** Using the products table from [mysqltutorial.org](https://www.mysqltutorial.org), write an SQL statement to get the number of Motorcycle product codes.

Analysis: This scenario counts the number of product codes belonging to the 'Motorcycles' product line.

The COUNT aggregate function is used to calculate the total count.

Table: products

Column: productLine

Condition: productLine = 'Motorcycles'

Aggregate Function: COUNT

Query:

```
SELECT COUNT(*) AS numberOfMotorcycleProducts FROM products WHERE productLine = 'Motorcycles';
```

SQL QUERY

1

2

```
SELECT COUNT(*) AS numberOfMotorcycleProducts FROM products WHERE productLine = 'Motorcycles';
```

▶ Execute

✕ Clear

✎ Beautify

✖ Minify

RESULT

numberOfMotorcycleProducts
13

**Scenario 8:** This scenario counts the number of employees with the job title 'Sales Rep'. The COUNT aggregate function is used to calculate the total count.

Analysis:

Table: employees

Column: jobTitle

Condition: jobTitle = 'Sales Rep'

Aggregate Function: COUNT

Query:

```
SELECT COUNT(*) AS numberOfSalesReps FROM employees WHERE jobTitle = 'Sales Rep';
```

SQL QUERY

```
1 SELECT COUNT(*) AS numberOfSalesReps FROM employees WHERE jobTitle = 'Sales Rep';
2
```

▶ Execute

✂ Clear

✍ Beautify

✖ Minify

RESULT

numberOfSalesReps
17

**Scenario 9:** Using the employees table from [mysqldata.org](https://mysqldata.org), write an SQL statement to get the number of employees for each position.

Analysis: This scenario counts the number of employees for each job position. The COUNT aggregate function is used, and the results are grouped by the jobTitle column.

Table: employees

Column: jobTitle

Aggregate Function: COUNT

Group By: jobTitle

Query:

```
SELECT jobTitle, COUNT(*) AS numberOfEmployees FROM employees GROUP BY jobTitle;
```



SQL QUERY

```
1 SELECT jobTitle, COUNT(*) AS numberOfEmployees FROM employees GROUP BY jobTitle;
2
```

▶ Execute

✎ Clear

✎ Beautify

✎ Minify

RESULT

jobTitle	numberOfEmployees
President	1
Sale Manager (EMEA)	1
Sales Manager (APAC)	1

**Scenario 10:** Using the orders table from [mysqltutorial.org](https://www.mysqltutorial.org), write an SQL statement to show the various statuses in the orders table and how many occurrences for each status.

Analysis: This scenario retrieves various order statuses and counts the occurrences of each status.

The COUNT aggregate function is used, and the results are grouped by the status column.

Table: orders

Column: status

Aggregate Function: COUNT


Group By: status

Query:

```
SELECT status, COUNT(*) AS statusCount FROM orders GROUP BY status;
```

## SQL QUERY

```
1 SELECT status, COUNT(*) AS statusCount FROM orders GROUP BY status;  
2
```

 Execute Clear Beautify Minify

## RESULT

status	statusCount
Cancelled	6
Disputed	3
In Process	6