📘 Module 20: Aggregate Commands

Aggregate functions perform **calculations on multiple rows** and return a **single value** (e.g., sum, count, average, etc.).

🔢 1. `COUNT()` – Count Records

✅ Definition:

Returns the **number of rows** or **non-null values** in a column.

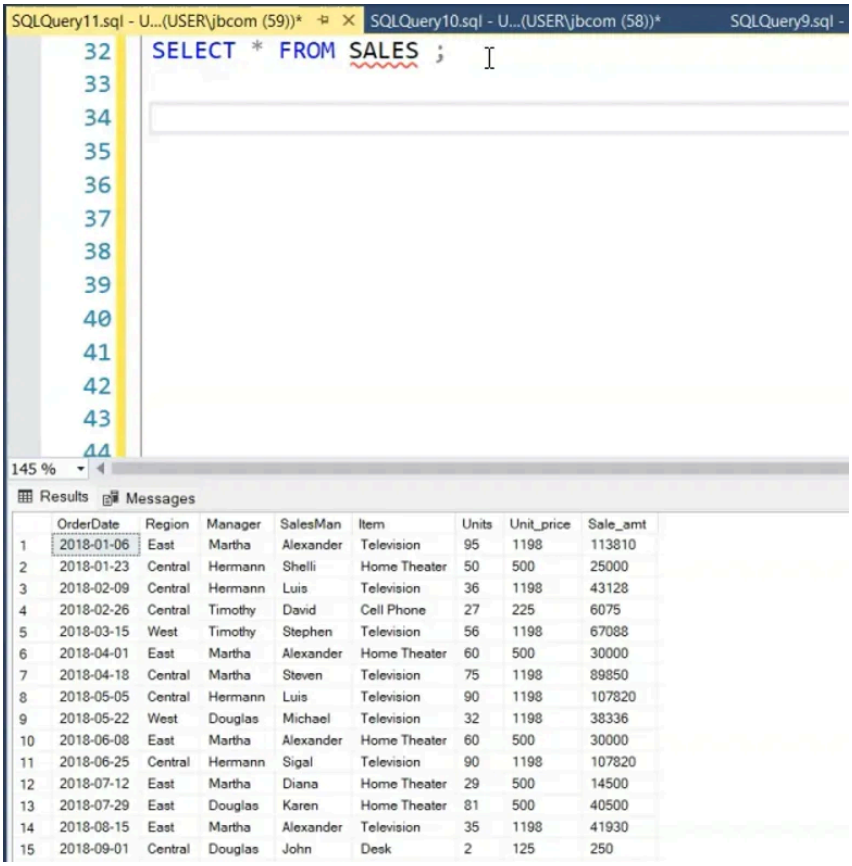
✅ Syntax:

```
SELECT COUNT(*) FROM table_name;
SELECT COUNT(column_name) FROM table_name WHERE condition;
```

💡 Why Use Aliasing?

- Makes the output column **meaningful**
- Helps **label the result** in reports or dashboards

🧾 Reference Table Used in This Module

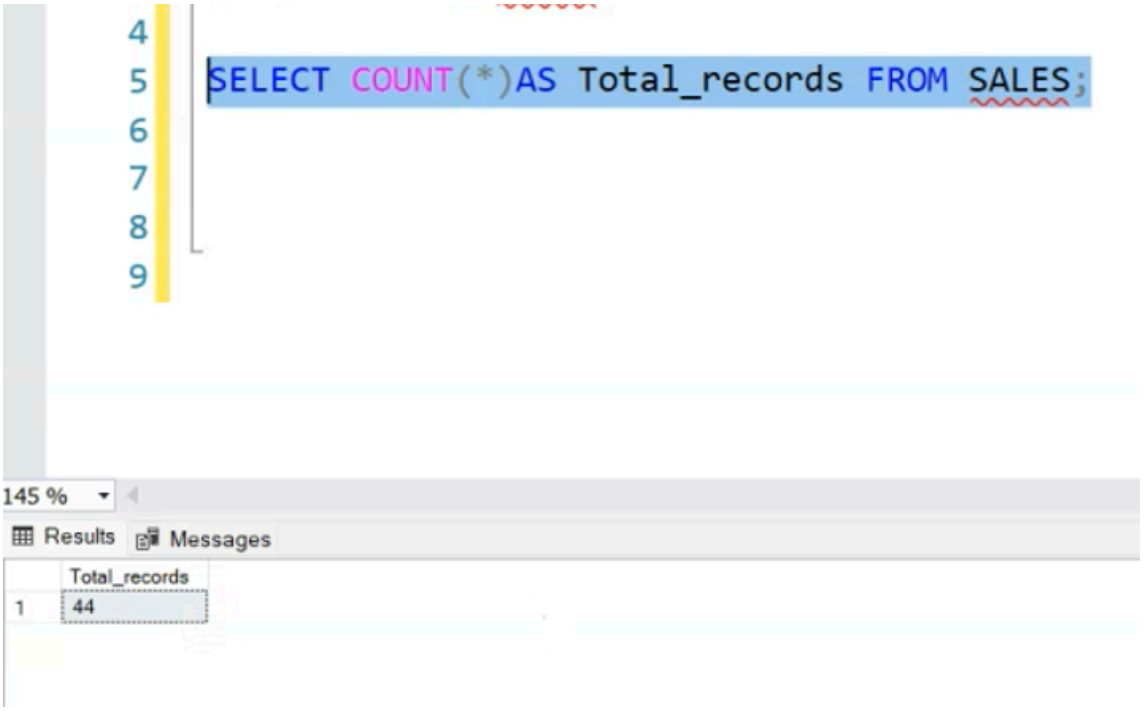


	OrderDate	Region	Manager	SalesMan	Item	Units	Unit_price	Sale_amt
1	2018-01-06	East	Martha	Alexander	Television	95	1198	113810
2	2018-01-23	Central	Hermann	Shelli	Home Theater	50	500	25000
3	2018-02-09	Central	Hermann	Luis	Television	36	1198	43128
4	2018-02-26	Central	Timothy	David	Cell Phone	27	225	6075
5	2018-03-15	West	Timothy	Stephen	Television	56	1198	67088
6	2018-04-01	East	Martha	Alexander	Home Theater	60	500	30000
7	2018-04-18	Central	Martha	Steven	Television	75	1198	89850
8	2018-05-05	Central	Hermann	Luis	Television	90	1198	107820
9	2018-05-22	West	Douglas	Michael	Television	32	1198	38336
10	2018-06-08	East	Martha	Alexander	Home Theater	60	500	30000
11	2018-06-25	Central	Hermann	Sigal	Television	90	1198	107820
12	2018-07-12	East	Martha	Diana	Home Theater	29	500	14500
13	2018-07-29	East	Douglas	Karen	Home Theater	81	500	40500
14	2018-08-15	East	Martha	Alexander	Television	35	1198	41930
15	2018-09-01	Central	Douglas	John	Desk	2	125	250

📌 Examples:

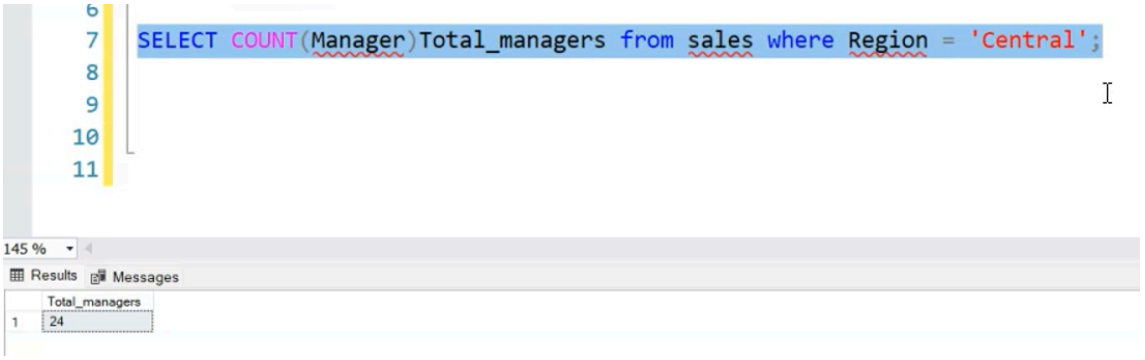
SELECT COUNT(*) AS Total_records FROM Sales;

✔️ Counts all rows in the Sales table.



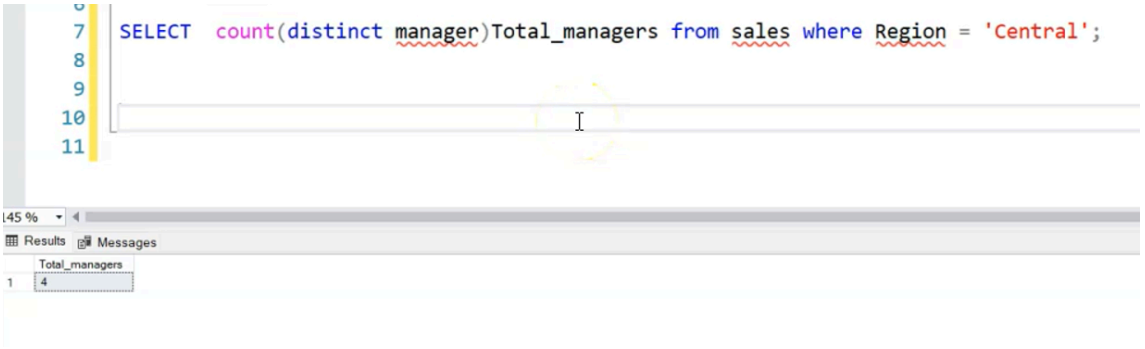
SELECT COUNT(Manager) AS Total_managers
FROM Sales
WHERE Region = 'Central';

✔️ Counts rows where the Manager is not NULL in the Central region.



SELECT COUNT(DISTINCT Manager) AS Unique_managers
FROM Sales
WHERE Region = 'Central';

✔️ Counts unique managers from the Central region.



➕ 2. `SUM()` – Total of Values

✅ Definition:

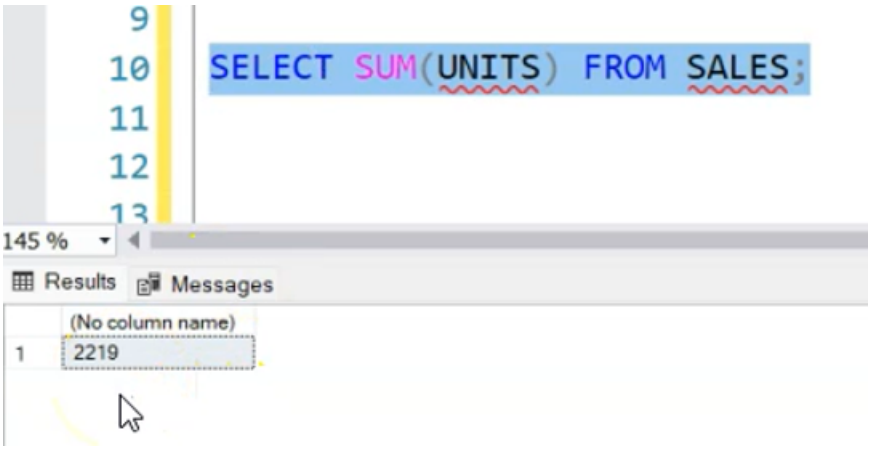
Returns the **sum of numeric values** in a column.

✅ Syntax:

SELECT SUM(column_name) FROM table_name WHERE condition;

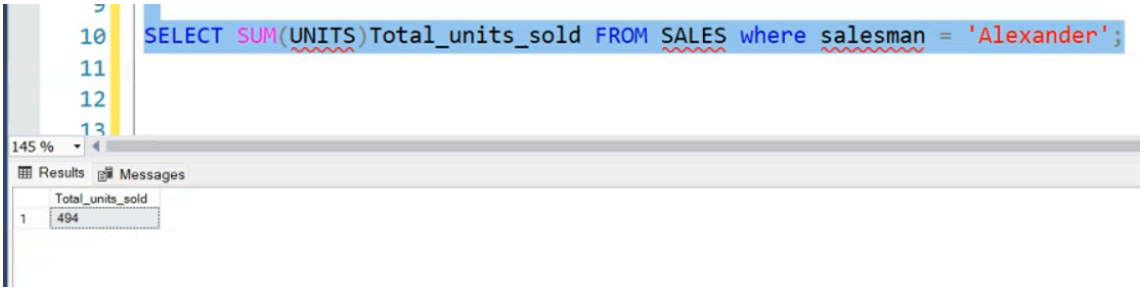
📌 Examples:

SELECT SUM(Units) FROM Sales;
✔️ Adds up all values in the
`Units` column.



SELECT SUM(Units) AS Total_units_sold
FROM Sales
WHERE SalesMan = 'Alexander';

✔️ Adds `Units` sold by **Alexander** only.



📊 3. `AVG()` – Average Value

✅ Definition:

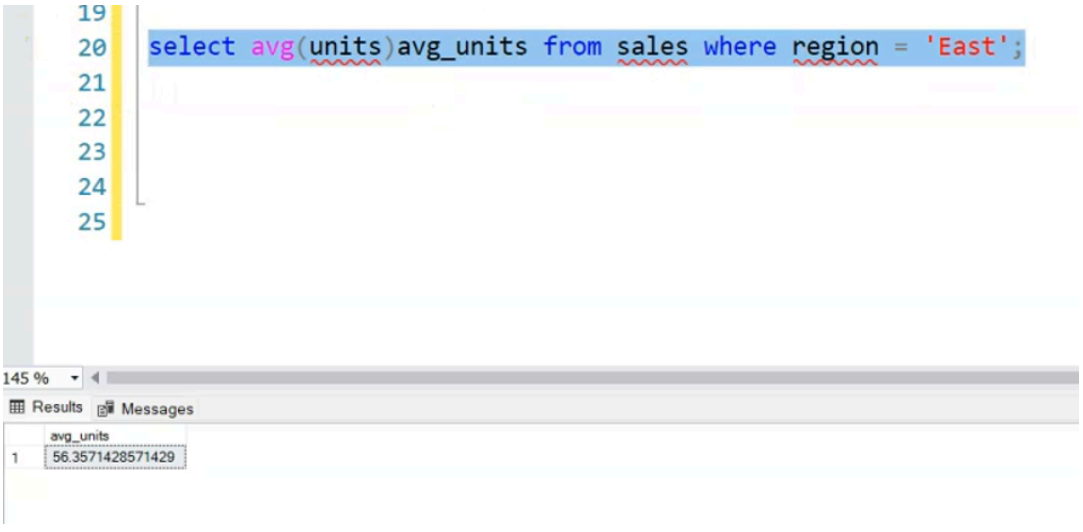
Returns the **average (mean)** of numeric values.

✅ Syntax:

SELECT AVG(column_name) FROM table_name WHERE condition;

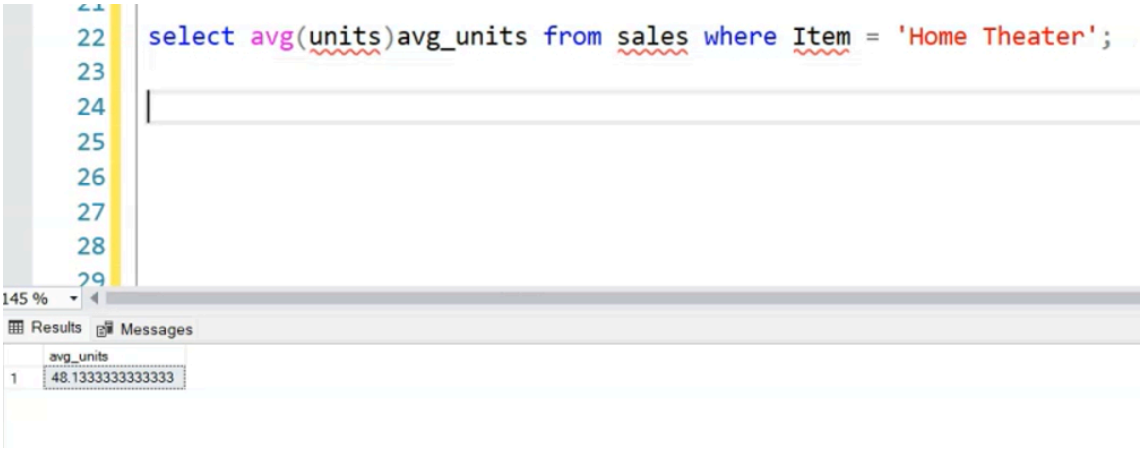
📌 Examples:

SELECT AVG(Units) AS Avg_Units
FROM Sales WHERE Region = 'East';
✔️ Averages
`Units` where Region = 'East'.



```
SELECT AVG(Units) AS Avg_Units
FROM Sales
WHERE Item = 'Home Theater';
```

✔️ Average
`Units` for all rows where Item = 'Home Theatre'.



📈 4. `MIN()` and `MAX()` – Minimum & Maximum

✅ Definition:

- `MIN()` returns the **lowest** value.
- `MAX()` returns the **highest** value.

✅ Syntax:

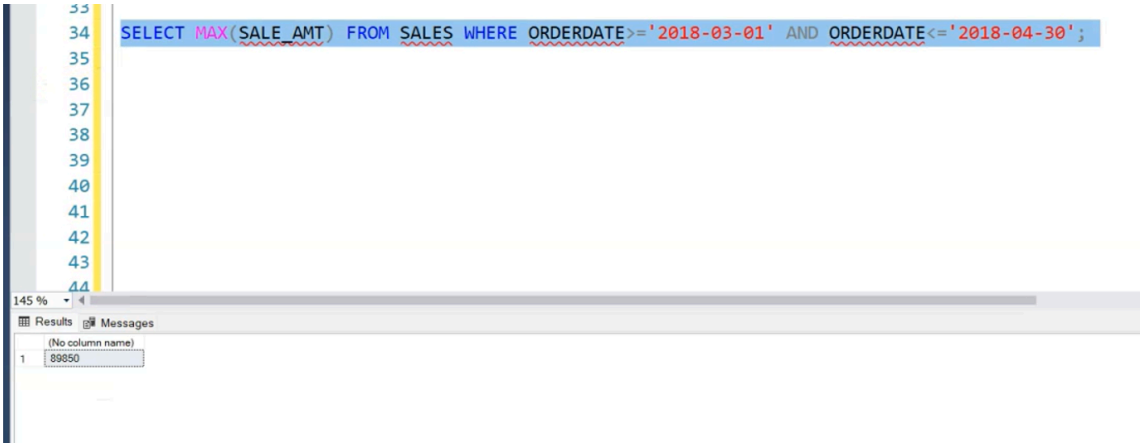
```
SELECT MIN(column_name) FROM table_name WHERE condition;
SELECT MAX(column_name) FROM table_name WHERE condition;
```

📌 Examples:

1.

```
SELECT MAX(Sale_amt) AS Max_Sales
FROM Sales
WHERE OrderDate BETWEEN '2018-03-01' AND '2018-04-30';
```

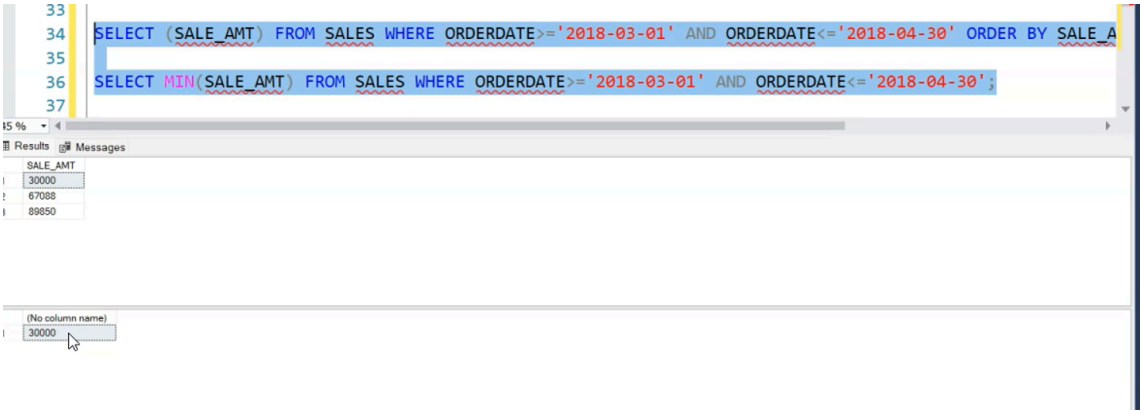
✔️ Gets the
maximum Sale_amt during March–April 2018.



2.

SELECT Sale_amt FROM Sales
WHERE OrderDate BETWEEN '2018-03-01' AND '2018-04-30'
ORDER BY Sale_amt ASC;
✔️ Lists all sale amounts within the specified date range,
sorted from lowest to highest.

SELECT MIN(Sale_amt) AS Min_Sales FROM Sales
WHERE OrderDate BETWEEN '2018-03-01' AND '2018-04-30';
✔️ Gets the
lowest Sale_amt during March–April 2018.



✔️**Similarity**: Both filter data between the same OrderDate range and work on the Sale_amt column.

✔️**Difference**: One shows **all values sorted**(Order By), while the other shows **only the minimum** using MIN().

✅ Summary of Aggregate Functions

Function	Description	Example
`COUNT()`	Count rows or values	`COUNT(*)` , `COUNT(DISTINCT column)`
`SUM()`	Add total values	`SUM(Units)`
`AVG()`	Calculate average	`AVG(Sale_amt)`
`MAX()`	Highest value	`MAX(Unit_price)`
`MIN()`	Lowest value	`MIN(Unit_price)`