

Counts and Totals

INTERMEDIATE SQL SERVER



Ginger Grant
Instructor

Examining Totals with Counts

```
SELECT COUNT(*) FROM Incidents
```

```
+-----+  
| (No column name) |  
+-----+  
| 6452              |  
+-----+
```

COUNT with DISTINCT

```
COUNT(DISTINCT COLUMN_NAME)
```

COUNT with DISTINCT in T-SQL (I)

```
SELECT COUNT(DISTINCT Country) AS Countries  
FROM Incidents
```

```
+-----+  
|Countries|  
+-----+  
|3        |  
+-----+
```

COUNT with DISTINCT in T-SQL (II)

```
SELECT COUNT(DISTINCT Country) AS Countries,  
COUNT(DISTINCT City) AS Cities  
FROM Incidents
```

```
+-----+-----+  
|Countries| Cities|  
+-----+-----+  
|3        | 3566  |  
+-----+-----+
```

COUNT AGGREGATION

- `GROUP BY` can be used with `COUNT()` in the same way as the other aggregation functions such as `AVG()` , `MIN()` , `MAX()`
- Use the `ORDER BY` command to sort the values
 - `ASC` will return the smallest values first (default)
 - `DESC` will return the largest values first

COUNT with GROUP BY in T-SQL

```
-- Count the rows, subtotaled by Country
SELECT COUNT(*) AS TotalRowsbyCountry, Country
FROM Incidents
GROUP BY Country
```

```
+-----+-----+
|TotalRowsbyCountry | Country      |
+-----+-----+
| 5452              | us           |
| 750               | NULL         |
| 249               | ca           |
| 1                 | gb           |
+-----+-----+
```

COUNT with GROUP BY and ORDER BY in T-SQL

(I)

```
-- Count the rows, subtotaled by Country
SELECT COUNT(*) AS TotalRowsbyCountry, Country
FROM Incidents
GROUP BY Country
ORDER BY Country ASC
```

```
+-----+-----+
|TotalRowsbyCountry | Country      |
+-----+-----+
| 750              | NULL        |
| 249              | ca          |
| 1                | gb          |
| 5452             | us          |
+-----+-----+
```


COUNT with GROUP BY and ORDER BY in T-SQL (II)

```
-- Count the rows, subtotaled by Country
SELECT COUNT(*) AS TotalRowsbyCountry, Country
FROM Incidents
GROUP BY Country
ORDER BY Country DESC
```

```
+-----+-----+
|TotalRowsbyCountry | Country      |
+-----+-----+
| 5452              | us           |
| 1                 | gb           |
| 249               | ca           |
| 750               | NULL        |
+-----+-----+
```

Column totals with SUM

- `SUM()` provides a numeric total of the values in a column
- It follows the same pattern as other aggregations
- Combine it with GROUP BY to get subtotals based on columns specified

Adding column values in T-SQL

```
-- Calculate the values subtotaled by Country
SELECT SUM(DurationSeconds) AS TotalDuration, Country
FROM Incidents
GROUP BY Country
```

```
+-----+-----+
|Country  |TotalDuration  |
+-----+-----+
|us       |17024946.750001565 |
|null     |18859192.800000012 |
|ca       |200975          |
|gb       |120             |
+-----+-----+
```

Let's practice!

INTERMEDIATE SQL SERVER

Math with Dates

INTERMEDIATE SQL SERVER



Ginger Grant
Instructor

DATEPART

`DATEPART` is used to determine what part of the date you want to calculate. Some of the common abbreviations are:

- `DD` for Day
- `MM` for Month
- `YY` for Year
- `HH` for Hour

Common date functions in T-SQL

- `DATEADD()` : Add or subtract datetime values
 - Always returns a date
- `DATEDIFF()` : Obtain the difference between two datetime values
 - Always returns a number

DATEADD

To Add or subtract a value to get a new date use `DATEADD()`

```
DATEADD (DATEPART, number, date)
```

- `DATEPART` : Unit of measurement (DD, MM etc.)
- `number` : An integer value to add
- `date` : A datetime value

Date math with DATEADD (I)

What date is 30 days from June 21, 2020?

```
SELECT DATEADD(DD, 30, '2020-06-21')
```

```
+-----+
| (No Column Name) |
|-----+
| 2020-07-21 00:00  |
+-----+
```

Date math with DATEADD (II)

What date is 30 days before June 21, 2020?

```
SELECT DATEADD(DD, -30, '2020-06-21')
```

```
+-----+
| (No Column Name) |
|-----+
| 2020-05-22 00:00  |
+-----+
```

DATEDIFF

Returns a date after a number has been added or subtracted to a date

```
DATEDIFF (datepart, startdate, enddate)
```

- `datepart` : Unit of measurement (DD, MM etc.)
- `startdate` : The starting date value
- `enddate` : An ending datetime value

Date math with DATEDIFF

```
SELECT DATEDIFF(DD, '2020-05-22', '2020-06-21') AS Difference1,  
       DATEDIFF(DD, '2020-07-21', '2020-06-21') AS Difference2
```

```
+-----+-----+  
|Difference1      |Difference2      |  
+-----+-----+  
|30              |-30              |  
+-----+-----+
```

Let's practice!

INTERMEDIATE SQL SERVER

Rounding and Truncating numbers

INTERMEDIATE SQL SERVER



Ginger Grant
Instructor

Rounding numbers in T-SQL

```
ROUND(number, length [,function])
```

Rounding numbers in T-SQL

```
SELECT DurationSeconds,  
ROUND(DurationSeconds, 0) AS RoundToZero,  
ROUND(DurationSeconds, 1) AS RoundToOne  
FROM Incidents
```

| DurationSeconds | RoundToZero | RoundToOne |
|-----------------|-------------|------------|
| 121.6480 | 122.0000 | 121.6000 |
| 170.3976 | 170.0000 | 170.4000 |
| 336.0652 | 336.0000 | 336.1000 |
| ... | | |

Rounding on the left side of the decimal

```
SELECT DurationSeconds,  
ROUND(DurationSeconds, -1) AS RoundToTen,  
ROUND(DurationSeconds, -2) AS RoundToHundred  
FROM Incidents
```

| DurationSeconds | RoundToTen | RoundToHundred |
|-----------------|------------|----------------|
| 121.6480 | 120.0000 | 100.0000 |
| 170.3976 | 170.0000 | 200.0000 |
| 336.0652 | 340.0000 | 300.0000 |
| ... | | |

Truncating numbers

TRUNCATE

17.85 \rightarrow 17

ROUND

17.85 \rightarrow 18

Truncating with ROUND()

The `ROUND()` function can be used to truncate values when you specify the third argument

```
ROUND(number, length [,function])
```

- Set the third value to a non-zero number

Truncating in T-SQL

```
SELECT Profit,  
ROUND(DurationSeconds, 0) AS RoundingtoWhole,  
ROUND(DurationSeconds, 0, 1) AS Truncating  
FROM Incidents
```

| Profit | RoundingtoWhole | Truncating |
|---------|-----------------|------------|
| 15.6100 | 16.0000 | 15.0000 |
| 13.2444 | 13.0000 | 13.0000 |
| 17.9260 | 18.0000 | 17.0000 |
| ... | | |

Truncating just cuts all numbers off after the specified digit

Let's practice!

INTERMEDIATE SQL SERVER

More math functions

INTERMEDIATE SQL SERVER



Ginger Grant
Instructor

Absolute value

Use `ABS()` to return non-negative values

```
ABS(number)
```

Using ABS in T-SQL (I)

```
SELECT ABS(-2.77), ABS(3), ABS(-2)
```

```
+-----+-----+-----+
|(No column name)| |(No column name)| |(No column name)|
+-----+-----+-----+
|2.77          |3          |2          |
+-----+-----+-----+
```


Using ABS in T-SQL (II)

```
SELECT DurationSeconds, ABS(DurationSeconds) AS AbsSeconds
FROM Incidents
```

```
+-----+-----+
|DurationSeconds|AbsSeconds|
+-----+-----+
|-25.36         |25.36     |
|-258482.44     |258482.44 |
|45.66          |45.66     |
+-----+-----+
```

Squares and square roots in T-SQL

```
SELECT SQRT(9) AS Sqrt,  
       SQUARE(9) AS Square
```

```
+-----+-----+  
|Sqrt   |Square   |  
+-----+-----+  
|3      |81       |  
+-----+-----+
```

Logs

- `LOG()` returns the natural logarithm
- Optionally, you can set the base, which if not set is 2.718281828

```
LOG(number [, Base])
```

Calculating logs in T-SQL

```
SELECT DurationSeconds, LOG(DurationSeconds, 10) AS LogSeconds
FROM Incidents
```

```
+-----+-----+
|DurationSeconds|LogSeconds|
+-----+-----+
|37800          |4.577491799837225|
|5              |0.6989700043360187|
|20             |1.301029995663981|
|...           |
+-----+-----+
```

Log of 0

You cannot take the log of 0 as it will give you an error

```
SELECT LOG(0, 10)
```

An invalid floating point operation occurred.

Let's practice!

INTERMEDIATE SQL SERVER