# Creating temporary variables

-- Creating temporary variables

-- Initialise a variable, give it a data type and an initial value

DECLARE @myvar as int = 2

-- Increase that value by 1

SET @myvar = @myvar + 1

-- Retrieve that value

SELECT @myvar AS myVariable

# Integer numbers

-- Initialise a variable, give it a data type and an initial value

DECLARE @myvar as smallint = 2000

-- Multiply that variable by 10

SET @myvar = @myvar \* 10

-- Retrieve that variable

SELECT @myvar AS myVariable

--BITS

--Bigint – 8 bytes

--Int - up to 2,000,000,000 4 bytes

--Tinyint - 0-255 – 1 bytes

--Smallint - -32767 to 32768 – 2 bytes

# Non-integer numbers

-- Initialise a variable, give it a data type and an initial value

DECLARE @myvar as numeric(7,2) -- or decimal(7,2) - 5 bytes needed

-- 12345.67 is valid for the above.

-- 123456.7 is not valid

SET @myvar = 12345.67

SELECT @myvar AS myVariable

GO

DECLARE @myvar as numeric(18,8) -- or decimal(18,8) - 9 bytes needed

SET @myvar = 1000000000.12345678

SELECT @myvar AS myVariable -- gives 12346

-- 1,000,000,000.12345678 is numeric(18,8)

GO

DECLARE @myvar AS smallmoney = 123456.78917 --4 bytes bigmoney= 8 bytes

--That's right - smallmoney can only store up to +/-214,748.364 or so.

select @myvar as myVariable -- gives 123456.7892

GO

DECLARE @myvar AS float(24) = 123456.7891 -- precise to 7 digits. -- same as REAL

Select @myvar as myVariable -- this gives 123456.8

# Mathematical functions

--Initialise a variable, give it a data type and an initial value

DECLARE @myvar as numeric(7,2) = 3

SELECT POWER(@myvar,3) -- 27

SELECT SQUARE(@myvar) -- 9

SELECT POWER(@myvar,0.5) -- square root of 3

SELECT SQRT(@myvar) -- square root of 3

GO

DECLARE @myvar as numeric(7,2) = 12.345

SELECT FLOOR(@myvar) -- this equals 12

SELECT CEILING(@myvar) -- this equals 13

SELECT ROUND(@myvar,-1) as myRound -- this equals 10 -1=10, 0=12, 1=12.40, 2=12.35

GO

SELECT PI() as myPI

SELECT EXP(1) as e

DECLARE @myvar AS NUMERIC(7,2) = -456

SELECT ABS(@myvar) as myABS, SIGN(@myvar) as mySign -- This equals 456 and -1.

GO

SELECT RAND(345) -- A random number, based on the initial seed

# Converting between number and types

-- IMPLICIT

DECLARE @myvar as Decimal(5,2) = 3

SELECT @myvar

-- explicit

SELECT CONVERT(decimal(5,2),3)/2

SELECT CAST(3 as decimal(5,2))/2

SELECT CONVERT(decimal(5,2),1000) -- this does not work

SELECT 3/2 -- EQUALS 1

SELECT 3/2.0 -- EQUALS 1.5

SELECT CONVERT(INT,12.345)+CONVERT(INT,12.7) -- This equals 24.

SELECT CONVERT(INT,12.345+12.7) -- This equals 25.

Enter the following SQL code:

* 1. select system\_type\_id, column\_id, system\_type\_id / column\_id as Calculation
  2. from sys.all\_columns

Please try the following questions. If you need a hint, then there is a hint document attached to these questions.

1. Have a look at the Calculation column. What is wrong with it? Please correct it.
2. Please round the fractions in the Calculation column down to the next whole number (so 4.153 would round down to 4).
3. Please round the fractions up (so 4.153 would round up to 5).
4. Please round the fractions to the nearest one decimal place (so 4.153 would round up to 4.2).
5. Multiply the first field, system\_type\_id, by 2, and then convert it to a tiniyint. If it doesn't work, instead of converting it using CONVERT or CAST, use the functions TRY\_CONVERT or TRY\_CAST instead - these give a NULL if the conversion doesn't work properly.

# Strings

-- char - ASCII - 1 byte

-- varchar - ASCII - 1 byte

-- nchar - UNICODE - 2 bytes

-- nvarchar - UNICODE - 2 bytes

-- ASCII: A-Z, a-z, 0-9

-- 65 97 48 32 128 (0-255)

DECLARE @chrMyCharacters as char(10)

set @chrMyCharacters = 'hello'

SELECT @chrMyCharacters as myString, len(@chrMyCharacters) as MyLength, DATALENGTH(@chrMyCharacters) as MyDataLength

-- Always prefix nchar and nvarchar values with N (capital N). if not the 'helloϞ' will give hello?

That's correct - the N refers to the NVARCHAR (or NCHAR), and it uses apostrophes to surround the value.

Yes - it is an NVARCHAR. It is 1 byte for each letter of hello, then multiply by 2 for the "N", plus 2 bytes because it is a "VAR". This makes (5\*2)+2 = 12.

# String Functions

DECLARE @chrASCII as varchar(10) = 'hellothere'

DECLARE @chrUNICODE as nvarchar(10) = N'helloϞ' -- datalength = 2\*10

select left(@chrASCII,2) as myASCII, right(@chrUNICODE,2) as myUNICODE -- he , oϞ

select substring(@chrASCII,3,2) as middleletters -- indexstart from 1

select ltrim(rtrim(@chrASCII)) as myTRIM --

As mentioned in the previous video, LTRIM removes space from the beginning of a string, and RTRIM removes spaces from the end.

As of SQL Server 2017, there is a new function - TRIM - which does both. However, don't use this on SQL Server 2016 or earlier, as it won't work.

select replace(@chrASCII,'l','L') as myReplace

select upper(@chrASCII) as myUPPER

select lower(@chrASCII) as myLOWER

# NULL – an introduction

declare @myvar as int

select 1+1+1+1+1+@myvar+1+1 as myCol -- NULL

declare @mystring as nvarchar(20)

select datalength(@mystring) as mystring

declare @mydecimal decimal(5,2)

select try\_convert(decimal(5,2),1000)

select try\_cast(1000 as decimal(5,2))

# Joining two strings together

declare @firstname as nvarchar(20)

declare @middlename as nvarchar(20)

declare @lastname as nvarchar(20)

set @firstname = 'Sarah'

--set @middlename = 'Jane'

set @lastname = 'Milligan'

-- select @firstname + ' ' + @middlename + ' ' + @lastname as FullName

select @firstname + iif(@middlename is null, '', ' ' + @middlename) + ' ' + @lastname as FullName

select @firstname + CASE WHEN @middlename IS NULL THEN '' ELSE ' ' + @middlename END + ' ' + @lastname as FullName

select @firstname + coalesce(' ' + @middlename,'') + ' ' + @lastname as FullName

SELECT CONCAT(@firstname,' ' + @middlename, ' ' , @lastname) as FullName

# Joining a string to a number

SELECT 'My number is: ' + convert(varchar(20),4567)

SELECT 'My number is: ' + cast(4567 as varchar(20))

SELECT 'My salary is: $' + convert(varchar(20),2345.6) -- works , but not well

SELECT 'My salary is: ' + format(2345.6,'C','fr-FR')

# Setting dates and date extraction

declare @mydate as datetime = '2015-06-24 12:34:56.124'

select @mydate as myDate

declare @mydate2 as datetime2(3) = '20150624 12:34:56.124' // precise till5 digit

select @mydate2 as MyDate

select DATEFROMPARTS(2015,06,24) as ThisDate

select DATETIME2FROMPARTS(2015,06,24,12,34,56,124,5) as ThatDate

select year(@mydate) as myYear, month(@mydate) as myMonth, day(@mydate) as myDay

# More date functions

SELECT CURRENT\_TIMESTAMP as RightNow

select getdate() as RightNow

select SYSDATETIME() AS RightNow

select dateadd(year,1,'2015-01-02 03:04:05') as myYear

select datepart(hour,'2015-01-02 03:04:05') as myHour

select datename(weekday, getdate()) as myAnswer

select datediff(second,'2015-01-02 03:04:05',getdate()) as SecondsElapsed

# Date offsets

declare @myDateOffset as datetimeoffset(2) = '2015-06-25 01:02:03.456 +05:30' -- 8-10 bytes

select @myDateOffset as MyDateOffset

go

declare @myDate as datetime2 = '2015-06-25 01:02:03.456'

select TODATETIMEOFFSET(@myDate,'+05:30') as MyDateOffset

select DATETIME2FROMPARTS (2015,06,25,1,2,3,456, 3)

select DATETIMEOFFSETFROMPARTS(2015,06,25,1,2,3,456,5,30,3) as MyDateOffset

select SYSDATETIMEOFFSET() as TimeNowWithOffset;

select SYSUTCDATETIME() as TimeNowUTC;

declare @myDateOffset as datetimeoffset = '2015-06-25 01:02:03.456 +05:30'

select SWITCHOFFSET(@myDateOffset,'-05:00') as MyDateOffsetTexas

# Converting from dates to strings

declare @mydate as datetime = '2015-06-25 01:02:03.456'

select 'The date and time is: ' + @mydate

go

declare @mydate as datetime = '2015-06-25 01:02:03.456'

select 'The date and time is: ' + convert(nvarchar(20),@mydate,104) as MyConvertedDate

go

declare @mydate as datetime = '2015-06-25 01:02:03.456'

select cast(@mydate as nvarchar(20)) as MyCastDate

select try\_convert(date,'Thursday, 25 June 2015') as MyConvertedDate

select parse('Thursday, 25 June 2015' as date) as MyParsedDate

select parse('Jueves, 25 de junio de 2015' as date using 'es-ES') as MySpanishParsedDate

select format(cast('2015-06-25 01:02:03.456' as datetime),'D') as MyFormattedLongDate

select format(cast('2015-06-25 01:02:03.456' as datetime),'d') as MyFormattedShortDate

select format(cast('2015-06-25 01:59:03.456' as datetime),'dd-MM-yyyy') as MyFormattedBritishDate

select format(cast('2015-06-25 01:02:03.456' as datetime),'D','zh-CN') as MyFormattedInternationalLongDate

Precident

Date

Number

String

So if num+ string it will try to convert string to num

* [How to add or subtract days or time in SQL Server using the DATEADD Function](https://www.youtube.com/watch?v=DYCWOzzOycU)
* [What is the difference between two dates? Using the DATEDIFF function](https://www.youtube.com/watch?v=J66l4V1Cwh4)