SQL Server Data Types & Type Casting

1) Write the SQL SELECT statement that returns the FirstName column of Person.Person casted as the VARCHAR data type.
2) Write three expressions in a single SELECT statement: one that returns the results of 11 divided by 4. The second column should return the result of 11 casted as float divided by 4 casted as float. The final column should divide 11.0 by 4.0 (including the decimal point and trailing zero).
3) Cast the FirstName column of Person.Person as the VARCHAR(3) data type. What happens? Why?
4) Many of the values in the Size column of the Production.Product table contain numeric values. Write a SELECT statement that returns the Size column casted as the integer data type. What is the result? Why?
5) Using the same SELECT statement that you developed in problem 4, add the WHERE clause, "WHERE ISNUMERIC(Size) = 1". What is the result of the query now? Why? (Hint: use the MSDN articles to find how the ISNUMERIC() function is used).

SQL Server Data Types & Type Casting Answer

Question 1:

```
SELECT
	CAST(FirstName AS VARCHAR)
FROM Person.Person
```

Question 2:

```
SELECT
	11/4,
	CAST(11 AS FLOAT)/CAST(4 AS FLOAT),
	11.0/4.0
```

Question 3:

```
SELECT
	CAST(FirstName AS VARCHAR(3))
FROM Person.Person
```

Question 4:

```
SELECT
	CAST(Size AS INT)
FROM Production.Product
```

An error is returned stating “conversion failed when converting the nvarchar value ‘M’ to data type int.”. This is because there are some values of the Size column that contain sizes like “S”, “M”, “L”, etc. These values are not numeric and cannot be converted to an integer as there is certainly no integer equivalent for the value of the letter “S” or “M”.

Question 5:

```
SELECT
	CAST(Size AS INT)
FROM Production.Product
WHERE ISNUMERIC(Size) = 1
```

With the addition of the WHERE clause and the appropriate filtering condition, we do not receive an error this time. The ISNUMERIC() function looks at a value and outputs a 1 if the value is numeric and a 0 if the value is not numeric. By specifying that we only want to evaluate numeric values, we bypass the error where we cannot convert non-numeric value to an integer. Since the WHERE clause is evaluated before the SELECT clause by the database, the CAST() function will not evaluate any values like “S”, “M”, or “L” which caused the errors in the previous question.